НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
(повна назва інституту/факультету)

Автоматизованих систем обробки інформації і управління
(повна назва кафедри)

«На правах рукопису»
УДК *004.4*

До захисту допущено:
В.о. завідувача кафедри
_____ Олександр ПАВЛОВ
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Інженерія програмного забезпечення комп'ютеризованих систем»

зі спеціальності 121 «Інженерія програмного забезпечення»

на тему: «*Інтелектуальна система розпізнавання мовлення у вивченні іншомовних слів на основі машинного навчання***»**

Виконав (-ла):
студент (-ка) VI курсу, групи ІП-92мп
Худа Анна Олександрівна _____

Науковий керівник:
Старший викладач кафедри АСОІУ
Халус Олена Андріївна _____

Рецензент:
К. т. н., доцент кафедри АУТС
Писаренко Андрій Володимирович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Автоматизованих систем обробки інформації і управління

Рівень вищої освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма – «Інженерія програмного забезпечення комп'ютеризованих систем»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Олександр ПАВЛОВ
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Худа Анні Олександрівні

1. Тема дисертації «*Інтелектуальна система розпізнавання мовлення у вивченні іншомовних слів на основі машинного навчання*», науковий керівник дисертації Халус Олена Андріївна, старший викладач кафедри АСОІУ, затверджені наказом по університету від «_26_»_жовтня_ 2020 р. №_3132-с

2. Термін подання студентом дисертації 17 грудня 2020 р.

3. Об'єкт дослідження:___*Процес визначення правильності вимови шляхом порівняння аудіофайлів.*

4. Перелік завдань, які потрібно розробити: *провести аналіз існуючих рішень даної проблеми та наявних аналогів; розробити архітектуру програмного забезпечення для постановки вимови; реалізувати зчитування звуку з мікрофону користувача та збереження його в аудіофайл; реалізувати генерацію аудіофайлу з тексту за допомогою технології Text-to-Speech; реалізувати алгоритм попередньої обробки аудіофайлів для їх синхронізації;реалізувати алгоритм порівняння двох аудіофайлів для виявлення помилки у вимові; проаналізувати результати роботи системи.*

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: *схема структурна бази даних, схема структурна бізнес-процесу визначення правильності вимови*.

7. Орієнтовний перелік публікацій: *публікація матеріалів роботи в збірнику тез IV всеукраїнської науково-практичної конференції молодих вчених та студентів «Інформаційні системи та технології управління» (ІСТУ-2020).*

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Графічний	доц. Ліщук К.І.		

9. Дата видачі завдання: 1 вересня 2020 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	*Отримання завдання*	*1.09.2020*	
2	*Аналіз існуючих технічних рішень та аналогів*	*2.09.2020 – 20.09.2020*	
3	*Вибір інструментів розробки*	*21.09.2020 – 30.09.2020*	
4	*Інтеграція існуючих методів синтезу мовлення*	*1.10.2020 – 8.10.2020*	
5	*Розробка системи розпізнавання мовлення*	*9.10.2020 – 23.10.2020*	
6	*Розробка алгоритму порівняння аудіофайлів*	*24.10.2020 – 7.11.2020*	
7	*Аналіз результатів*	*8.11.2020 – 13.11.2020*	
8	*Оформлення документації*	*14.11.2020 – 19.11.2020*	
9	*Розробка стартап-проекту*	*20.11.202 – 23.11.2020*	
10	*Подання роботи на попередній захист*	*27.11.2020*	
11	*Подання роботи на основний захист*	*17.12.2020*	

Студент Анна ХУДА

Науковий керівник Олена ХАЛУС

РЕФЕРАТ

Магістерська дисертація:

Актуальність теми. Сьогодні знання іноземних мов, а тим більше англійської, є неохідним. Якщо ж мову можна вивчити за допомогою різноманітних онлайн курсів, то ситуація з вимовую є складнішою. Є декілька варіантів тренування вимови. Перший варіант – слухати та повторювати. Цей варіант не є дуже ефективним, оскільки людині складно почути свою вимову зі сторони, а тим більше оцінити її. Другий спосіб – курси іноземних мов. Цей варіант ефективний, але доволі дорогий. Третій підхід – використання спеціалізованих застосувань для спілкування з носіями мови для її вивчення. Третій варіант є дуже ефективним, але має декілька недоліків. По перше, складно знайти людину яка дійсно зацікавлена у тому, щоб допомогти з постановкою вимови, оскільки такі застосунки, в основному, призначені для того, щоб люди просто спілкувалися на різноманітні теми і не замислювались над постановкою вимови. По друге, є люди, яким некомфортно спілкуватись з незнайомими людьми і це є значною перешкодою для вивчення мови за допомогою таких застосунків.

Таким чином вирішення задачі постановки вимови за допомогою інтелектуальної системи, яка має можливість аналізувати вимову користувача є дуже актуальною в наш час.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалась на кафедрі автоматизованих систем обробки інформації та управління факультету обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» в рамках теми «Інтелектуальна система розпізнавання мовлення у вивченні іншомовних слів на основі машинного навчання»

Метою дослідження є покращення процесу постановки вимови шляхом розробки архітектури програмного забезпечення, що буде порівнювати два аудіофайли для перевірки правильності вимови іншомовних слів. Перший аудіофайл зчитується з мікрофону користувача, другий генерується системою за

допомогою технології Text-to-Speech.

Для досягнення поставленої мети необхідно виконати наступні завдання:

– провести аналіз існуючих рішень даної проблеми та наявних аналогів;

– розробити архітектуру програмного забезпечення для постановки вимови;

– реалізувати зчитування звуку з мікрофону користувача та збереження його в аудіофайл;

– реалізувати генерацію аудіофайлу з тексту за допомогою технології Text-to-Speech;

– реалізувати генерацію текста з аудіофайла за допомогою технології Speech-to-Text;

– реалізувати алгоритм попередньої обробки аудіофайлів для їх синхронізації;

– реалізувати алгоритм порівняння двох аудіофайлів для виявлення помилки у вимові;

– проаналізувати результати роботи системи.

Об’єкт дослідження. Процес визначення правильності вимови шляхом порівняння аудіофайлів.

Предметом дослідження є алгоритми і методи для розпізнавання мовлення, а також методи для порівняння аудіофайлів.

Наукова новизна одержаних результатів. Найбільш суттєвими науковими результатами магістерської дисертації є:

– запропоновано архітектурне рішення для побудови інтелектуальної системи розпізнавання мовлення для постановки вимови;

– вперше розроблена система постановки вимови, що, на відміну від подібних систем, є агностичною до кількості та набору слів та фраз, а також є точнішою, оскільки порівняння виконується безпосередньо на аудіофайлах.

Практичне значення отриманих результатів. Розроблена система використовується у застосуванні для вивчення іноземних слів для постановки вимови.

Публікації. Матеріали роботи опубліковані в збірнику тез IV всеукраїнської науково-практичної конференції молодих вчених та студентів «Інформаційні системи та технології управління» (ІСТУ-2020) Худа А. О., Халус О. А. «Метод порівняння аудіофайлів у системі для вивчення іноземних слів».

РОЗПІЗНАВАННЯ МОВЛЕННЯ, ІНОЗЕМНІ МОВИ, ПОРІВНЯННЯ АУДІО, TEXT-TO-SPEECH, SPEECH-TO-TEXT.

ABSTRACT

Master's Thesis:

Actuality. Today, knowledge of foreign languages, especially English, is essential. If the language can be learned with the help of various online courses, the situation with pronunciation is more complicated. There are several pronunciation training options. The first option is to listen and repeat. This option is not very effective, because it is difficult for a person to hear his pronunciation from the side, much less evaluate it. The second way is foreign language courses. This option is effective, but quite expensive. The third approach is to use specialized applications to communicate with native speakers to study it. The third option is very effective, but has several disadvantages. First of all, it is difficult to find a person who is really interested in helping with pronunciation, because such applications are mainly intended for people to simply communicate on various topics and not think about pronunciation. Secondly, there are people who are uncomfortable communicating with strangers and this is a significant obstacle to learning a language through such applications.

Thus, solving the problem of pronunciation with the help of an intelligent system that has the ability to analyze the user's pronunciation is very relevant today.

Relationship with working with scientific programs, plans, topics. The work was performed at the Department of Automated Information Processing and Management Systems of the Faculty of Computer Engineering of The National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" within the work “Intelligent speech recognition system in the process of learning foreign words based on machine learning”.

The aim of the study is to develop a software architecture that will compare two audio files to check the correct pronunciation of foreign words. The first audio file is read from the user's microphone, the second is generated by the system using Text-to-Speech technology.

To achieve this goal it is necessary to perform the following tasks:

– to analyze the existing solutions of this problem and existing analogues;

– develop a software architecture for pronunciation improvement;

– implement reading the sound from the user's microphone and saving it to an

audio file;

– implement the generation of an audio file from text using Text-to-Speech technology;

– implement the generation of text from an audio file using Speech-to-Text technology;

– implement an algorithm for pre-processing audio files for their synchronization;

– implement an algorithm for comparing two audio files to detect an error in pronunciation;

– analyze the results of the system.

Object of study. The process of determining the correct pronunciation by comparing audio files.

The subject of the research are re algorithms and methods for speech recognition, as well as methods for comparing audio files.

Scientific novelty of the obtained results. The most significant scientific results of the master's dissertation are:

– an architectural solution is proposed to build an intelligent speech recognition system for pronunciation;

– the first developed pronunciation system, which, unlike similar systems, is agnostic to the number and set of words and phrases, as well as more accurate, because the comparison is performed directly on audio files.

The practical significance of the results. The developed system is used in the application for the study of foreign words for improving pronunciation.

Publications. Materials are published in the collection of abstracts of the IV All-Ukrainian scientific-practical conference of young scientists and students "Information systems and control technologies" (ISTU-2020) Khuda Anna, Khalus Olena "Method of comparing audio files in the system for learning foreign words" .

SPEECH RECOGNITION, FOREIGN LANGUAGES, AUDIO COMPARISONS, TEXT-TO-SPEECH, SPEECH-TO-TEXT.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

TTS – Text to Speech

DTW – Dynamic Time Wrapping

GRU – Gated recurrent Unit

MVC – Model View Controller

ВСТУП

При вивченні іноземних мов необхідно розвивати чотири види мовленнєвої діяльності: читання, письмо, слухання і говоріння. Якщо перші три вміння можна розвивати самостійно або за допомогою спеціальних програм, то у випадку говоріння – складніше, оскільки перевірити правильність вимови може викладач на курсах іноземної мови або носій іноземної мови. Але найефективнішим методом поповнення словникового запасу – є комплексний підхід до вивчення слів, який включає в себе усі названі вище види мовленевої діяльності.

Проблема розглянута в даній магістерській дисертації – постановка вимови при вивченні іншомовних слів.

Є декілька варіантів тренування вимови. Перший варіант – слухати та повторювати. Цей варіант не є дуже ефективним, оскільки людині складно почути свою вимову зі сторони, а тим більше оцінити її. Другий спосіб – курси іноземних мов. Цей варіант ефективний, але доволі дорогий. Третій підхід – використання спеціалізованих застосувань для спілкування з носіями мови для її вивчення. Третій варіант є дуже ефективним, але має декілька недоліків. По перше, складно знайти людину яка дійсно зацікавлена у тому, щоб допомогти з постановкою вимови, оскільки такі застосунки, в основному, призначені для того, щоб люди просто спілкувалися на різноманітні теми і не замислювались над постановкою вимови. По друге, є люди, яким некомфортно спілкуватись з незнайомими людьми і це є значною перешкодою для вивчення мови за домогою таких застосунків.

Таким чином вирішення задачі постановки вимови за допомогою інтелектуальної системи, яка має можливість аналізувати вимову користувача є дуже актуальною в наш час.

1 ПРОЕКТНІ РІШЕННЯ З РОЗРОБКИ СИСТЕМИ РОЗПІЗНАВАННЯ МОВЛЕННЯ ДЛЯ ВИВЧЕННЯ ІНОЗЕМНИХ СЛІВ

1.1 Опис бізнес процесів

1.1.1 Опис процесу діяльності

Об’єктом автоматизації є процес розпізнавання мовлення, генерація аудіофайлу з тексту та подальша обробка аудіофайлів для визначення правильності вимови.

Схема структурна бізнес-процесу визначення правильності вимови наведена у графічному матеріалі.

При натисканні користувачем кнопки початку завдання в застосунку для вивчення іноземних мов, сервер робить запит до бази даних для отримання текстового представлення слова і передає отримане слово на клієнта для відображення користувачу. Користувач натискає кнопку запису та вимовляє слово і після цього натискає кнопку зупинки запису. Тим часом на стороні сервера з текстового представлення слова генерується його звукове представлення за допомогою технології Text-to-Speech. Після того, як користувач зупинив запис клієнт передає запис на сервер, де відбувається порівняння двох аудіофайлів – зчитаного з мікрофону користувача і згенерованого системою. Якщо, при аналізі двох аудіофайлів виявляється, що вимова однакова кількість правильних відповідей збільшеється на 1. Якщо вимова не співпадає, кількість неправильних відповідей збільшується на 1 і після виконання завдання на сторінці результату відображаються слова, які були вимовлені неправильно. Кожне слово можна прослухати, а також для кожного слова відображається транскрипція з позначенням, який звук було вимовлено неправильно.

1.1.2 Варіанти використання

При користуванні системою користувач має можливість виконати завдання по постановці вимови та отримати результати проходження цього завдання. Неправильно вимовлені слова користувач має можливість прослухати після виконання задання, а також подивитись транскрипцію цих слів з зазначенням частини, в якій вимова була неправильна.

Взаємодія користувача і системи зображена рисунку 1.2 – Схема структурна варіантів використання.

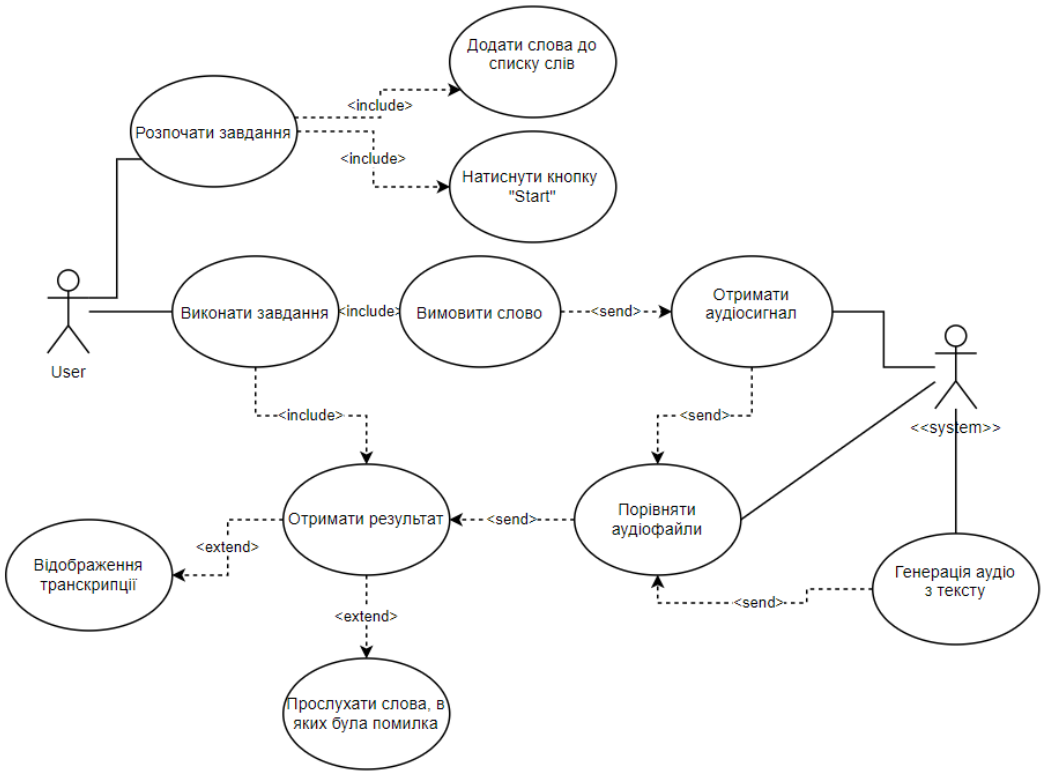


Рисунок 1.1 – Схема структурна варіантів використання

1.2 Постановка задачі

Метою дослідження є розробка системи, що буде порівнювати два аудіофайли для перевірки правильності вимови іншомовних слів. Перший аудіофайл зчитується з мікрофону користувача, другий генерується системою за допомогою технології Text-to-Speech.

Для досягнення поставленої мети необхідно виконати наступні завдання:

– провести аналіз існуючих рішень даної проблеми та наявних аналогів;

– реалізувати зчитування звуку з мікрофону користувача та збереження його в аудіофайл;

– реалізувати генерацію аудіофайлу з тексту за допомогою технології Text-to-Speech;

– реалізувати генерацію текста з аудіофайла за допомогою технології Speech-to-Text;

– реалізувати алгоритм попередньої обробки аудіофайлів для їх

синхронізації;

– реалізувати алгоритм порівняння двох аудіофайлів для виявлення помилки у вимові;

– проаналізувати результати роботи системи.

1.3 Наявні рішення

Основною задачею даної системи є синтез, розпізнавання та порівняння мовлення для постановки вимови при вивченні іноземних мов, зокрема англійської

1.3.1 Наявні технічні рішення для синтезу мовлення

При аналізі наявних технічних рішень для синтезу мовлення були обрані найпопулярніші Text-to Speech системи, а саме:

– Amazon Polly;

– Azure Text to Speech;

– Google Cloud Text-to-Speech;

– IBM Watson Text to Speech.

Розглянемо кожну із них та спробуємо виділити переваги та недоліки для визначення, яке з цих рішень доцільніше використовувати для вирішення задачі синтезу мовлення.

Amazon Polly - це послуга, яка перетворює текст на реалістичну мову, що дозволяє створювати програми, що розмовляють, і створювати абсолютно нові категорії мовних продуктів. Послуга Поллі з перетворення тексту в мовлення використовує передові технології глибокого навчання для синтезу людської мови, що звучить природно. За допомогою десятків реалістичних голосів на різних мовах ви можете створювати програми з підтримкою мови, які працюють у багатьох різних країнах.

В додаток до стандартних голосів TTS, Amazon Polly пропонує голоси нейронного перетворення тексту в мовлення (NTTS), які забезпечують вдосконалену якість мовлення завдяки новому підходу до машинного навчання. Технологія Neural TTS від Polly також підтримує два стилі мовлення, які дозволяють краще відповідати стилю передачі доповідача до застосування: стиль читання Newscaster, пристосований до випадків використання розповіді про

новини, та Conversational стиль розмови, ідеально підходить для двостороннього спілкування, такого як програми телефонії.

Основними перевагами Amazon Polly є:

– дана система підтримує понад 15 різних мов;

– користувачі Amazon Polly мають повний контроль над звучанням мови, від висоти до вимови;

– має підтримку платформування та мови програмування: вона підтримує усі мови програмування, що входять до набору інструментів Amazon Web Services SDK;

– користувачі мають можливість створювати власні лексикони.

Звісно, дана система має і недоліки:

– Amazon Polly не розпізнає деякі символи;

– не підтримує переклад між мовами;

– деякі голоси Amazon Polly все ще звучать трохи роботизовано.

Azure Text to Speech – TTS система від Microsoft, яка дозволяє застосункам, інструментам або пристроям перетворювати текст у синтезовану мову, схожу на людську. Можливий вибір серед більш ніж 75 стандартних голосів, які доступні більш ніж на 45 мовах. Також має 5 нейронних голосів. З допомогою Azure Text to Speech можна створити власний голос, унікальний для розроблюваного продукту чи бренду.

Основні переваги Azure Text to Speech:

– підтримка великої кількості мов;

– можливість створити унікальний голос.

Основні недоліки Azure Text to Speech:

– звук не дуже якісний;

– голос не дуже реалістичний;

– доволі складна інтеграція.

Google Cloud Text-to-Speech дозволяє розробникам синтезувати природнє звучання мови більш ніж 40-а мовами та більш ніж 220 голосами. Дана система застосовує новаторські дослідження DeepMind у WaveNet та потужних нейронних

мережах Google, щоб забезпечити високу якість звуку.

Основні переваги Google Cloud Text-to-Speech:

– висока якість мовлення;
– великий вибір голосу;
– можливість створити власний голос;
– інтонація, схожа на людську;
– проста інтеграція API.

При аналізі основних недоліків не було знайдено суттєвих недоліків.

IBM Watson Text to Speech – сервіс перетворення тексту в мовлення розуміє текст і природну мову для того, щоб генерувати синтезований аудіосигнал у комплекті з відповідною частотою та інтонацією. Синтез доступний 27 голосами (13 нейронних та 14 стандартних) на 7 мовах.

Основні переваги IBM Watson Text to Speech:

– проста інтеграція API;
– підтримка багатьох мов програмування.

Основні недоліки IBM Watson Text to Speech:

– підтримується не така велика кількість мов, як у конкурентів;
– вимова не завжди чітка;
– дорожчий ніж інші подібні рішення, розгялнуті вище.

1.3.2 Наявні технічні рішення для розпізнавання мовлення

При аналізі наявних технічних рішень для розпізнавання мовлення були обрані найпопулярніші Speech-to-Text системи, а саме:

– AssemblyAI - Speech to Text API;
– IBM Watson Speech to Text;
– Amazon Transcribe;
– Google Cloud Speech-to-Text.

AssemblyAI - Speech to Text API має точну транскрипцію звуку, використовуючи найновіші дослідження в сфері глибокого навчання. Однією з найбільш важливих особливостей AssemblyAI є застосування декількох різних бібліотек для різних акцентів, якості запису та середовища запису, враховуючи

кількість фонових шумів. Лише ця функція робить AssemblyAI вартою для розгляду, оскільки транскрипція звуку з динаміків із незнайомими акцентами може бути надзвичайно складною та трудомісткою.

Основні переваги AssemblyAI:

– досить висока точність і швидкість;
– приємна ціна;
– проста інтеграція API;
– підтримка багатьох мов програмування.

Основні недоліки AssemblyAI:

– немає точності рівня людини.

IBM Watson Speech to Text – хмарне рішення, яке використовує алгоритми глибокого навчання для застосування знань про граматику, структуру мови та композицію звукових сигналів для створення налаштовуваного розпізнавання мови для оптимальної транскрипції тексту.

Основні переваги IBM Watson Speech to Text:

– проста інтеграція API;
– точна інтерпретація речення та його контексту.

Основні недоліки IBM Watson Speech to Text:

– не фокусується на мові лише однієї людини. Якщо будь-яка мова розпізнається, IBM Watson Speech to Text намагається це перетворити в текст і це дуже впливає на якість розпізнавання;
– невелика кількість підтримуваних мов.

Amazon Transcribe – сервіс автоматичного розпізнавання мовлення (ASR), який полегшує розробникам впровадження Speech-to-Text у програмні застосунки. Використовуючи API Amazon Transcribe, можна аналізувати аудіофайли, що зберігаються в Amazon S3 і отримувати текстові файли транскрибованої мови.

Основні переваги Amazon Transcribe:

– підтримка багатьох аудіоформатів;
– можливість виокремити одного розмовника.

Основні недоліки Amazon Transcribe:

– розпізнавання дуже сильно залежить від контексту;

– якщо розмір аудіосигналу, що подається на вхід невеликий, час обробки доволі довгий.

Google Cloud Speech-to-Text застосовує найсучасніші алгоритми глибокого навчання нейронних мереж Google для автоматичного розпізнавання мови.

Основні переваги Google Cloud Speech-to-Text:

– підтримка 125 мов та варіантів вимови;

– можливість обрати навчену модель;

– проста інтеграція API.

При аналізі основних недоліків не було знайдено суттєвих недоліків.

1.3.3 Наявні аналоги

Було розглянуто декілька продуктів на ринку подібних застосувань і було визначено, що існують або застосування для вивчення слів, в яких немає завдань для постановки вимови або завстосування тільки для постановки вимови.

Застосування для вивчення слів, в основному, надають користувачу можливість вивчення слів за допомогою карток та завдання, які базуються на картках.

При аналізі застосувань для постановки вимови були обрані такі застосування для дослідження:

Utter. Інтерфейс програми Utter добре опрацьований і виглядає на професійному рівні. Після авторизації в застосуванні, користувач починає з курсу Tenses, в якому про програму розповідає бот зі штучним інтелектом. Він також тестує користувача через інтерактивний чат, в якому потрібно дати відповіді на кілька запитань. Це важливо для правильного налаштування програми. Існують різні курси для різних сценаріїв. Наприклад, можна обрати курс для спілкування з друзями про поїздки, офісі, роботі. Навчання проводиться на трьох рівнях:

- початковий - для повсякденних завдань таких, як запитати час або дізнатися дорогу;
- проміжний - для звичайної бесіди з друзями;
- розширений - для офісу, робочого спілкування.[1]

Після вивчення основ починається серйозне освоєння англійської вимови. Користувачу потрібно буде записувати свій голос, даючи відповіді на різні питання. На кожному рівні надається по два безкоштовних урока. Щоб продовжити навчання, доведеться заплатити $ 2,49. [1]

Elsa. ELSA (назва розшифровується як English Language Speech Assistant) – це додаток для інтерактивного спілкування. Він допомагає поліпшити вимову і в цілому почати краще говорити англійською. У ELSA є теми уроків для різних учнів - від водіїв таксі до співробітників служби клієнтської підтримки. У кожній темі є багато уроків. Коли користувач запускає один з них, програма відтворює слова і фрази. Користувач повинен повторювати їх. Користувач миттєво дізнається про оцінку своєї вимови. Залежно від результатів програма попросить або повторити слово або фразу, або дозволить рухатися далі. Якщо виникнуть труднощі зі словом, є можливість скористатися підказками щодо руху мови і губ. Потрібно натиснути на кнопку у вигляді вуха, щоб прослухати відтворення слова. Програма має вбудований словник на 2000 часто використовуваних слів і фраз. [1]

Nativox. Nativox фокусується на тому, щоб допомогти користувачам краще зрозуміти особливості усної американської англійської. При використанні Nativox користувач слухає всиловлювання і дивиться, як слова збираються в цікаву діаграму. Так простіше зрозуміти, як вимовляти кожну частину речення. Потім користувач зможе повторювати фрази і записувати свій голос. Можна імітувати вимову до тих пір, поки користувач не зможе максимально наблизитися до зразка. Nativox розроблений так, щоб користувач міг навчитися говорити щось і паралельно дізнаватися переклад слів. Так мозок може зосередитися на усному мовленні, а не на граматиці або лексиці. [1]

English Pronunciation. English Pronunciation - це маленький додаток, який не тільки підкаже, як правильно вимовляти слова, але і покаже як правильне положення органів дикції для цього. Запускаючи додаток в перший раз, потрібно вибрати свою рідну мову. Потім програма покаже, як вимовляти голосні і приголосні. Натиснувши на голосну або приголосну букву, користувач побачить розширені опції. На екрані буде показано положення мови і губ з детальним

текстовим поясненням. Також є різні приклади слів, які можна прослухати з британським або американським акцентом. У розділі «Practice» є слова з відсутніми літерами. Потрібно правильно підібрати букви і ввести все слово повністю. Є контрольні точки і відстеження прогресу, щоб полегшити процес навчання. На жаль, не можна просто набрати якесь слово щоб почути його вимову. Програма поширюється безкоштовно, хоча з рекламними оголошеннями і покупками через додаток. Можна отримати додаткові тести і приклади слів. [1]

1.4 Глибоке навчання

Глибоке навчання – це техніка машинного навчання, яка вчить комп'ютери робити те, що природньо для людини: вчитися на прикладі. Глибоке навчання – ключова технологія, що стоїть за автономними автомобілями, які вміють розпізнавати дорожні знаки, смуги руху або, наприклад, відрізняти пішохода від ліхтарного стовпа. Це ключ до голосового управління побутовими пристроями, такими як телефони, планшети, телевізори та гучномовці. Останнім часом глибоке навчання привертає багато уваги і з не з проста. Це досягнення результатів, які раніше були неможливі.

При глибокому навчанні комп'ютерна модель вчиться виконувати класифікаційні завдання безпосередньо із зображень, тексту чи звуку. Моделі глибокого навчання можуть досягти надсучасної точності, іноді перевищуючи показники на рівні людини. Моделі навчаються за допомогою великого набору маркованих даних та архітектур нейронних мереж, які містять багато шарів.

Глибоке навчання досягає точності розпізнавання вище, ніж будь-коли раніше. Це допомагає побутовій електроніці відповідати очікуванням користувачів, і це має вирішальне значення для програм, в яких на першому місці стоїть безпека, таких як автономні автомобілі. Останні досягнення глибокого навчання вдосконалились до такої міри, що глибоке навчання перевершує людей у деяких завданнях, таких як класифікація об'єктів на зображеннях.

Хоча глибоке навчання вперше було теоретизовано у 1980-х, є дві основні причини, чому воно стало нещодавно корисним:

– для глибокого навчання потрібні великі обсяги маркованих даних.

Наприклад, для розробки автомобілів без драйверів потрібні мільйони зображень і тисячі годин відео.

– Глибоке навчання вимагає значних обчислювальних потужностей. Високопродуктивні графічні процесори мають паралельну архітектуру, ефективну для глибокого навчання. У поєднанні з кластерами або хмарними обчисленнями це дозволяє розробникам скоротити час навчання моделі глибокого навчання з тижнів до годин або менше.

Більшість методів глибокого навчання використовують архітектури нейронних мереж, саме тому моделі глибокого навчання часто називають глибокими нейронними мережами. Термін "глибокий" зазвичай позначає кількість прихованих шарів у нейронній мережі. Традиційні нейронні мережі містять лише 2-3 прихованих шари, тоді як глибинні мережі можуть мати цілих 150.

Моделі глибокого навчання навчаються за допомогою великих наборів маркованих даних та архітектур нейронних мереж, які вивчають функції безпосередньо з даних без необхідності ручного вилучення функцій.

Існують три основні типи нейронних мереж, які складають основу для більшості попередньо підготовлених моделей глибокого навчання:

– нейронні мережі прямого поширення;

– згорткові нейронні мережі;

– рекурентні нейронні мережі.

Усі нейронні мережі складаються з трьох типів шарів, які зображені на рисунку 1.3 – шари нейронної мережі:

– вхідні;

– приховані;

– вихідні.

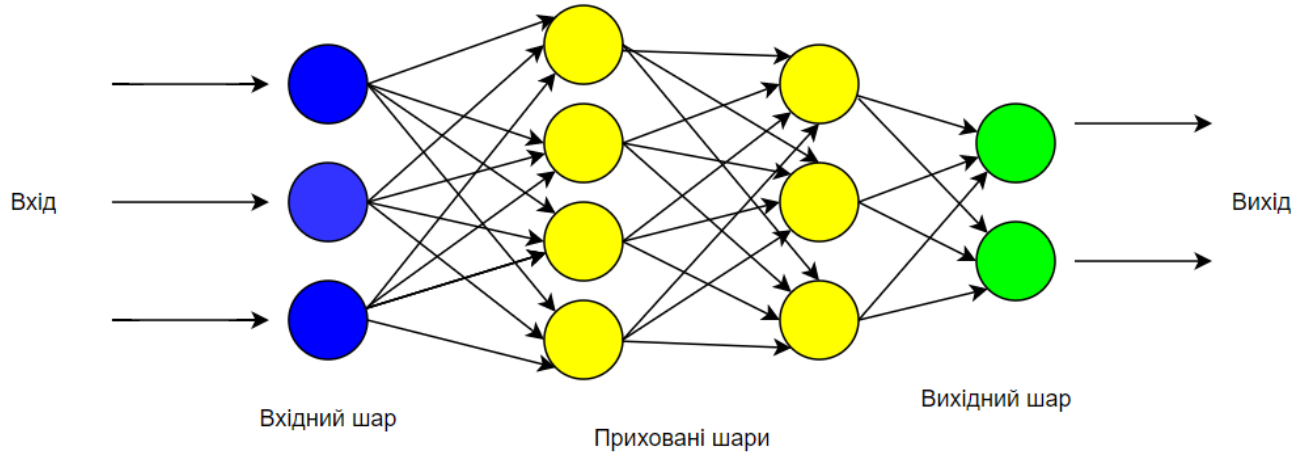


Рисунок 1.2 – Шари нейронної мережі

Нейронні мережі прямого поширення зазвичай використовуються для вирішення задач з такими типами даних, як табличні та текстові дані, а також з зображеннями. Рекурентні нейронні мережі використовуються для вирішення задач, пов'язаних з даними часових рядів, текстовими даними та аудіоданими. Згорткові нейронні мережі особливо поширені для вирішення задач, що пов'язані з обробкою зображень та відео.

Висновки до розділу

В даному розділі були розглянуті бізнес процеси системи розпізнавання мовлення для вивчення іноземних слів, була розроблена схема структурна бізнес процесів та схема структурна варіантів використання.

Були розглянуті найпопулярніші технічні рішення для синтезу мовлення, а саме Amazon Polly, Azure Text to Speech, Google Cloud Text-to-Speech, IBM Watson Text to Speech. Проаналізувавши їх переваги та недоліки, було обрано Google Cloud Text-to-Speech API для синтезу мовлення. Також, були розглянуті найпопулярніші технічні рішення для розпізнавання мовлення. Серед них також розглядались API від Amazon, Google та IBM. Було проведено аналіз їх переваг та недоліків і в результаті було обрано Google Cloud Speech-to-Text API для вирішення задачі розпізнавання мовлення. Поруч із наявними технічними рішеннями було проведено аналіз застосувань-аналогів і в результаті було виявлено, що точних аналогів розроблюваної системи для вивчення іноземних мов не існує.

В цьому розділі наведені теоретичні відомості з області глибокого навчання.

2 ДЕТАЛЬНИЙ ОГЛЯД ТЕХНОЛОГІЙ СИНТЕЗУ МОВИ ТА РОЗПІЗНАВАННЯ МОВЛЕННЯ

2.1 Технологія синтезу мови

Комп’ютерне мовлення існує вже досить давно. Однак якість генерованного мовлення все ще не достатньо схоже на людське і нелегко сприймається. Хоча подібні системи досить близько, вони не є достатньо приближеними, навіть системи від гігантів, таких як Google, Apple або Amazon, далекі від звучання, подібного до людського. У цьому розділі розглядається як останні досягнення у мовленнєвому синтезі використовують технології глибокого навчання для формування природнього звучання мови.

Щоб зрозуміти, чому сьогодні технології глибокого навчання використовуються для синтезу мови, важливо розуміти, як здійснюється генерація мовлення. Існує два специфічні методи перетворення тексту в мову (TTS): параметричний TTS та сполучний TTS. Важливо також визначити такі два терміни, щоб оцінити якість генерованої мови: зрозумілість і природність. Зрозумілість – це якість звуку, що генерується. Природність – це якість генерованого мовлення.

2.1.1 Сполучна TTS система

Як випливає з назви, ця техніка базується на високоякісних аудіо відрізках (або блоках) записів, які потім об’єднуються разом, щоб сформувати мовлення. Хоча генерований мовний звук дуже чистий і чіткий, він звучить без емоцій. Звук виходить зрозумілим, але звучить не природньо. Це пов’язано з тим, що доволі складно знайти аудіозаписи усіх можливих вимов слів, з урахуванням можливих поєднань емоцій, просодій, наголосів тощо. Звичайно, ці системи потребують величезних баз даних і вже закодованої комбінації для формування цих слів. Розробка надійної системи займає від семи до восьми місяців.

2.1.2 Параметрична TTS система

Сполучна TTS система є дуже обмеженою через високі вимоги до даних та час, виділений, на розробку. Саме тому був розроблений більш статистичний метод. Цей метод генерує мовлення, комбінуючи такі параметри, як основну частоту, величину спектру тощо та обробляє їх для генерації мови. Параметрична

система TTS має два етапи. На першому етапі виділяються мовні особливостей під час обробки тексту. Ці особливості можуть бути фонемами, тривалістю тощо. На другому етапі виділяються особливості вокодера, які представляють відповідний мовний сигнал. Цими особливостями можуть бути цепстра, спектрограма, основна частота тощо, вони представляють деякі, властиві людській мові, характеристики і використовуються в обробці звуку. Ці ручно параметри разом з лінгвістичними особливостями передаються у математичну модель, яка називається Вокодер. Вокодер приймає ці параметри та виконує кілька складних перетворень на цих параметрах, щоб згенерувати звукову форму сигналу. Генеруючи сигнал, вокодер оцінює такі особливості мови, як фаза, просодія (ритм і стрес), інтонація тощо.

Параметрично синтезоване мовлення є високомодульним і цілком здійсненним. Якщо є можливість наблизити параметри, що впливають на мовлення, то можна навчити модель генерувати всі види мови. І створення такої системи вимагає значно менших обсягів даних та роботи, ніж сполучна TTS система.

2.1.3 Синтез мовлення з використанням глибокого навчання

Моделі глибокого навчання виявились надзвичайно ефективними при вивченні характеристик, що властиві даним. Ці характеристики не сприймаються людиною, але сприймаються комп’ютером і набагато краще представляють дані для моделі. Іншими словами, модель глибокого навчання вивчає функцію, для співставлення входу X з виходом Y. Працюючи над цим припущенням, система переводу тексту в мову, що звучить природньо, повинна приймати на вхід X як рядок тексту, а на вихіді Y – форму звукової хвилі. Система не повинна використовувати будь-які розроблені вручну функції, а навпаки, вивчати нові багатовимірні ознаки, щоб відобразити те, що робить мовлення, людину.

Аудіофайли представлені в комп'ютері шляхом оцифрування звукового сигналу. Це, по суті, часовий ряд аудіо-зразків. Отже, замість того, щоб генерувати якийсь прихований параметр, а потім обробляти його, щоб отримати аудіо, є сенс безпосередньо генерувати аудіо-зразки. Новаторською роботою з генерації звуку на рівні вибірки з глибокими нейронними мережами є WaveNet від DeepMind.

2.1.4 WaveNet

WaveNet генерує окремий зразок для аудіо, і кожен зразок визначається всіма попередніми зразками, створеними в аудіо. Потім цей зразок використовується з попередніми зразками для створення наступного зразка. Це називається авторегресивним поколінням.

$$p(X) = \prod_{i=0}^{T-1} p(x_{i+1}|x_1, \dots, x_i)$$

WaveNet побудований з використанням стеків згорткових шарів із залишками та пропускає з'єднання між ними. В якості вхідного сигналу приймається оцифрований необроблений звуковий сигнал, який потім протікає через ці згорткові шари і виводить зразок сигналу.

Модель незалежна, що означає, що їй не надається жодна інформація про структуру мовлення, тому вона не генерує жодного значущого звуку. Якщо навчити дану модель на аудіо, на якому говорять люди, вона буде видавати звуки, які здаватимуться, ніби люди говорять деякі слова, але ці слова будуть схожими на бурмотіння. Після цього команда WaveNet надала моделі параметри вокодера, з вже існуючої TTS системи, на вхіду разом із вихідним аудіосигналом. В результаті, отримана система синтезувала високоякісний голос. Він був дуже чистий, без шуму, без незрозумілого буркотіння. Однак це обчислення було дуже дорогим. Звичайний WaveNet для високоякісної мови використовує 40 таких згорнутих шарів, а також інші з'єднання між ними. Оскільки за один раз генерується один зразок, то для генерації 1 секунди 16 кГц звуку потрібно обробити 16000 зразків. Команда WaveNet повідомила, що для створення 1 секунди звуку потрібно близько 4 хвилин.

2.1.5 SampleRNN

SampleRNN - це ще один підхід до генерації звукових сигналів, він використовує ієрархію повторюваних шарів, які мають різні тактові частоти для обробки аудіо. Ідея полягає в тому, що декілька мереж RNN з'єднані в ієрархію. Верхній рівень приймає великі шматки вхідних даних, обробляє їх і передає

нижньому рівню; нижній рівень приймає менші шматки вхідних даних і передає їх далі. Це продовжується до самого нижнього рівня в ієрархії, де генерується одна вибірка. На рисунку 2.1 – Трирівневий SampleRNN зображено приклад трирівневого SampleRNN

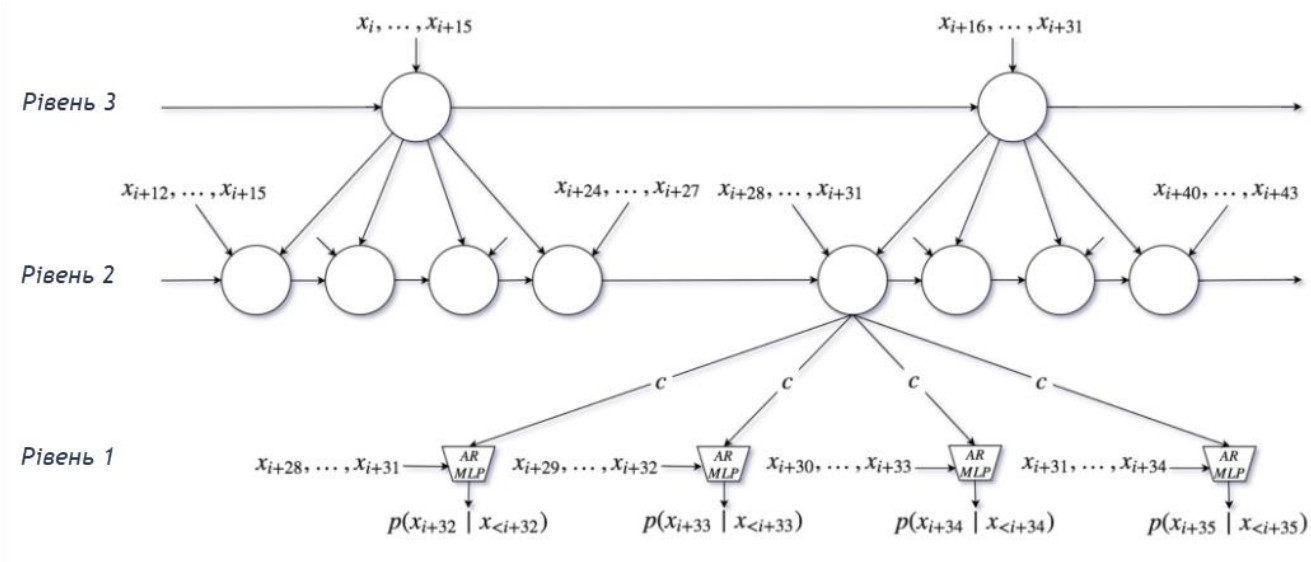


Рисунок 2.1 – Трирівневий SampleRNN

Як і WaveNet, це також є авторегресивною генеративною моделлю. Але обчислювальні дуже швидко в порівнянні з WaveNet.

2.1.6 Tacatron

Tacatron – неперервний синтезатор мовлення. Tacotron не робить припущення про те, які функції слід передавати вокодеру, як і не робить припущення про те, як слід обробляти текст. Команда Tacotron знає, що люди знають не все, і тому вони дозволяють моделі вивчати відповідні функції та обробку. Таким чином, Такотрон переходить на рівень символів. Він приймає символи тексту як вхідні дані, передає їх через різні підмодулі нейронної мережі та генерує спектрограму звуку. Це більше схоже на повну модель глибокого навчання для синтезу мовлення, і вона не вимагає жодних функцій від існуючих TTS систем, на відміну від WaveNet.

Модель Tacatron зображена на рисунку 2.2 – Tacatron архітектура.

Модель приймає символи як вхідні дані і виводить відповідну необроблену спектрограму, яка потім подається в алгоритм реконструкції Гріффіна-Ліма для

синтезу мовлення.

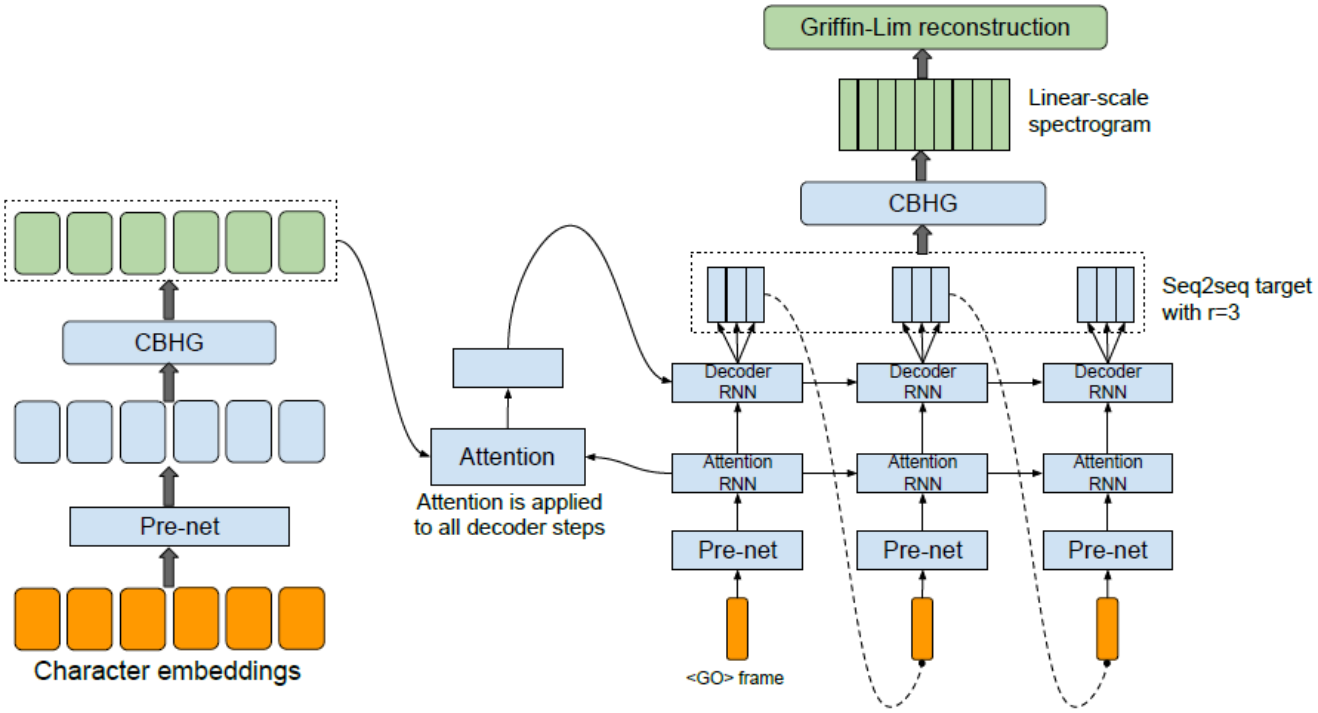


Рисунок 2.2 – Tacatron архітектура

Енкодер працює над інтеграцією символів вхідного тексту. Результат передається шару Pre-net та CBHG. Pre-Net – це ієрархічна паралельна рекурентна нейронна мережа. Після цього Tacotron енкодер використовує модуль під назвою CBHG поверх Pre-Net. Назва цього модуля походить від його будівельних блоків: одновимірний згортковий банк (1D Convolutional Bank – CB), після якого слідує шосейна мережа (highway network - H) та двонаправлений GRU (Gated recurrent Unit – G). GRU вивчає довгострокові залежності в послідовності. Модуль CBHG зображений на рисунку 2.3 – CBHG.

Декодер використовує attention механізм на результаті з енкодера для створення кадрів мел-спектрограми. Декодер також має Pre-net шар, за яким слідує одношаровий GRU, вихід якого об'єднаний з виходом енкодера для отримання контекстного вектора за допомогою attention механізму. Потім результат GRU об'єднується з контекстним вектором для отримання вхідного сигналу блоку декодера RNN.

Модуль RNN декодера створює *r*-число мел-спектрограмних кадрів, і лише останній використовується Pre-net шаром під час наступного кроку.

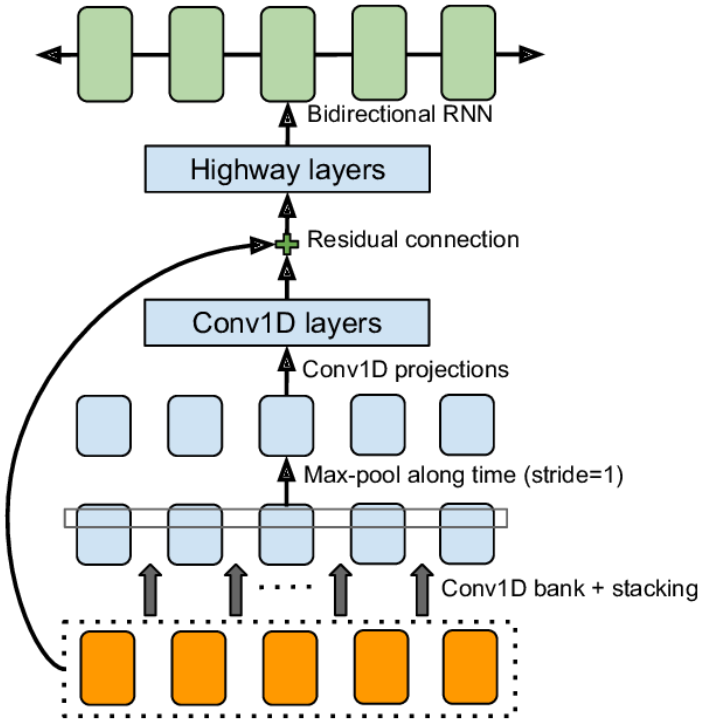


Рисунок 2.3 – CBHG

2.2 Розпізнавання мовлення

Розпізнавання мови вторгається в наше життя. Воно вже є в телефонах, ігрових консолях і навіть в смарт-годинах. Навіть будинки можна автоматизувати за допомогою мови. Але розпізнавання мови з'явилося десятиліття назад - так чому ж це тільки зараз воно стало настільки популярним? Причина в тому, що глибоке навчання нарешті зробило розпізнавання мови досить точним, щоб воно стало корисно поза ретельно контрольованим середовищем. У цьому підрозділі розглянемо як розпізнати мову за допомогою глибокого навчання.

Можна здогадатися, що можна просто згодувати нейромережі звукові записи і навчити її на них. Але серйозна проблема полягає в тому, що мова має непостійну швидкість. Одна людина може сказати слово, наприклад «привіт» досить швидко, тоді як інша скаже те ж саме слово дуже повільно, створивши набагато довший звуковий файл з набагато більшим обсягом даних. А між тим, обидва звукових файли повинні бути розпізнані як однаковий текст.

Перший крок в розпізнаванні мови очевидний – потрібно передати звуки на комп'ютер. Звук – це хвилі. Розглянемо як перетворити звукові хвилі в числа

Скористаємося аудіофайлом з текстом «привіт». Візуалізація даного аудіофайл представлена на рисунку 2.4 – Візуалізація аудіофайла.

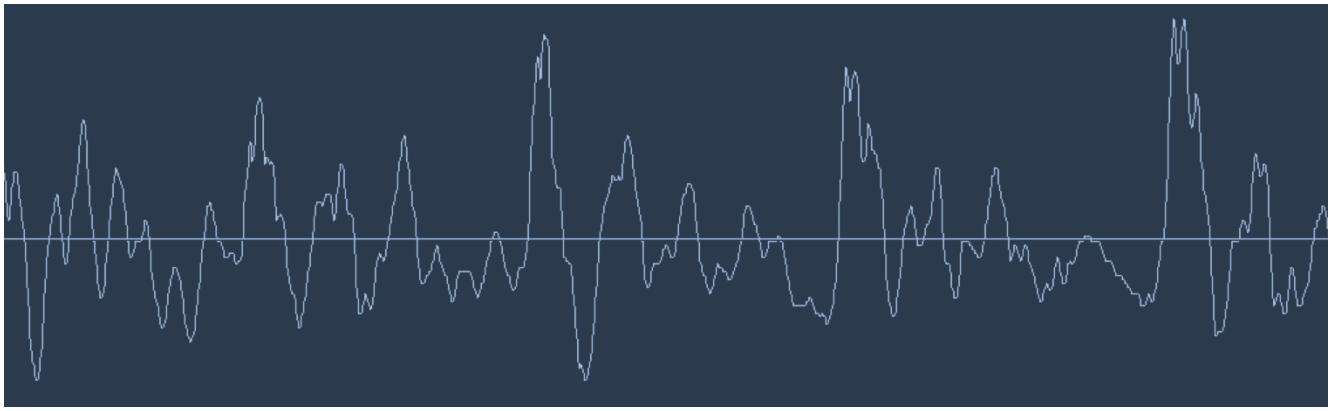


Рисунок 2.4 – Візуалізація аудіофайла

Звукові хвилі одномірні. У кожен момент часу у них є одне значення, залежне від амплітуди хвилі. Наблизимо деяку невелику частину звукової хвилі. Ця частина звукової хвилі наведена на рисунку 2.5 – Частина звукової хвилі.

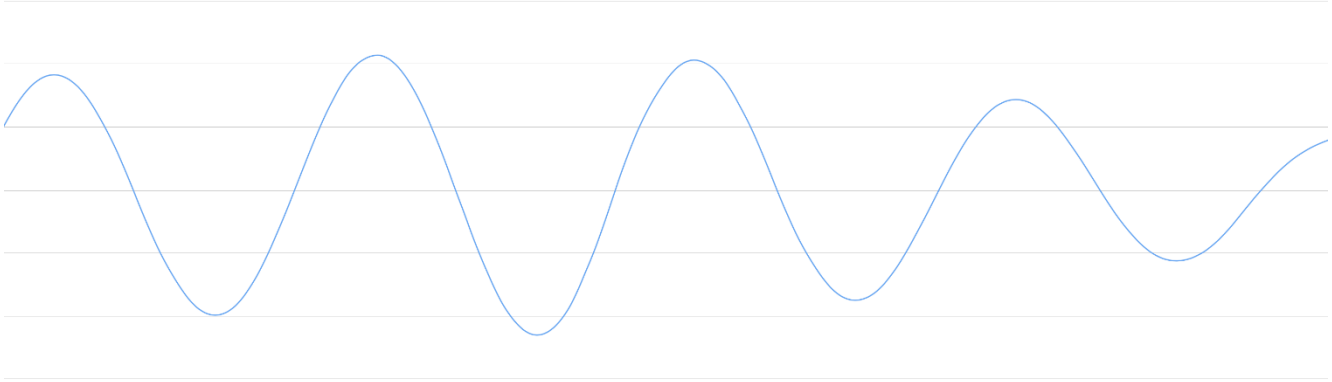


Рисунок 2.5 – Частина звукової хвилі

Для перетворення звукової хвилі в числа просто запишемо значення амплітуди хвилі в рівновіддалених точках (рисунок 2.6 – Амплітуда хвилі в рівновіддалених точках)

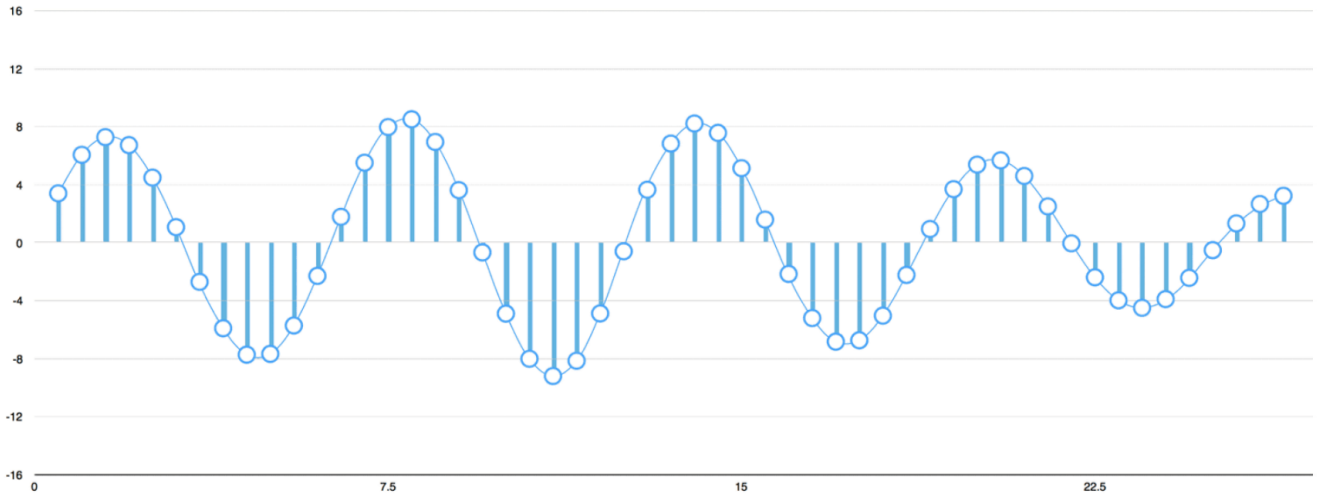


Рисунок 2.6 – Амплітуда хвилі в рівновіддалених точках

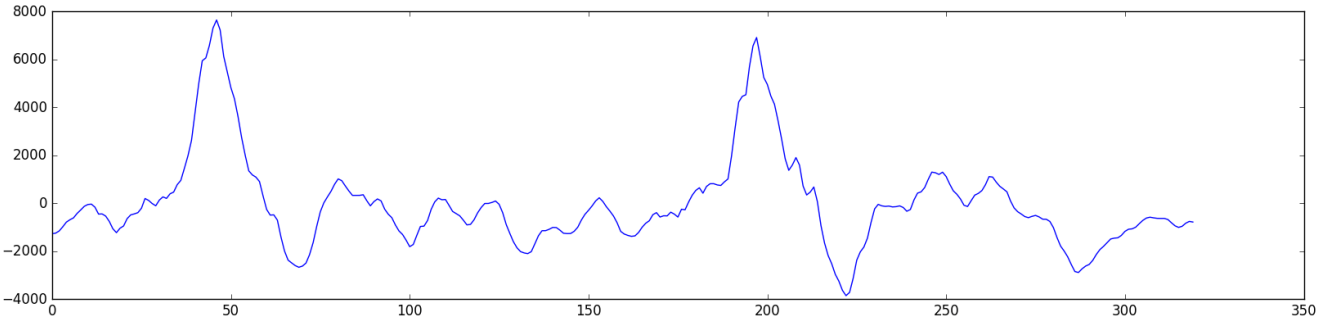
Даний процес має назву дискретизація. Дані зчитуються тисячі разів на секунду і записуються числа, що відповідають амплітуді звукової хвилі на той момент. В результаті отримуються стиснуті аудіофайли у форматі .wav.

Звук, записаний на компакт-дисках, має частоту дискретизації рівну 44,1 кГц (44 100 семплів в секунду). Оскільки діапазон частот людського мовлення не такий широкий, для його розпізнавання достатньо частоти дискретизації 16 кГц (16 000 семплів на секунду).

Може здатися, що дискретизація створює лише приблизну версію оригінальної звукової хвилі, оскільки ми зчитуємо випадкові значення, а в проміжках між відліками дані втрачаються. Завдяки теоремі Котельникова відомо, що для найкращого відтворення оригінальної звукової хвилі достатньо використати частоту дискретизації, яка вдвічі більша за найвищу частоту записаного звуку.

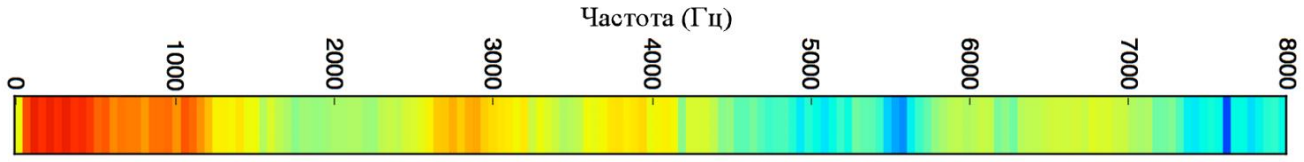
Далі необхідно оцифрувати слово «Привіт» 16000 раз в секунду. В результаті отримаємо набір чисел, кожне з яких відображає амплітуду звукової хвилі з інтервалом 1/16000 секунди. Можна було б вже просто навчити нейромережу на отриманих числах, але розпізнання лінгвістичних закономірностей шляхом безпосередньої обробки цих чисел важко. Натомість можна спростити завдання, попередньо обробивши аудіоінформацію. Почнемо з групування показань у 20 мілісекундні фрагменти. Побудова цих чисел у вигляді простого лінійного графіка,

що зображений на рисунку 2.7 – Приблизне зображення звукової хвилі за 20 мілісекунд, забезпечує приблизне зображення вихідної звукової хвилі для обраного періоду часу в 20 мілісекунд.



Рисунку 2.7 – Приблизне зображення звукової хвилі за 20 мілісекунд

Наведений фрагмент триває лише 1/50 секунди. Але це теж складна суміш різних частот звуку. Є кілька низькочастотних звуків, є середні звуки і деякі високі звуки також. Всі ці частоти змішані між собою, в результаті виходить звук людського мовлення. Розіб’ємо складну звукову хвилю на її складові, починаючи від низьких частот, для спрощення обробки цих даних нейронною мережою. Потім, додаючи потужності звуку в кожній смузі частот, створимо частотну картину звуку. Це робиться за допомогою математичної операції, званої перетворенням Фур'є. Робиться розбивання складної звукової хвилі на прості звукові хвилі, з яких вона складається. Отримавши окремі звукові хвилі, складемо потужності звуку в кожній із них. В результаті отримаємо оцінку важливості кожного частотного діапазону, від низьких частот до високих. На діаграмі, що зображена на рисунку 2.8 – Діаграма частот, зображена потужність звуку в кожній смузі по 50 Гц у оригінальному фрагменті.



Рисунку 2.8 – Діаграма частот

Якщо повторити цей процес на кожному 20-мілісекунди фрагменті аудіо, отримаємо спектрограму. В спектрограмі, зображеній на рисунку 2.9 –

Спектрограма аудіофайла, кожен стовпець зліва направо є одним 20-мілісекунди фрагментом.

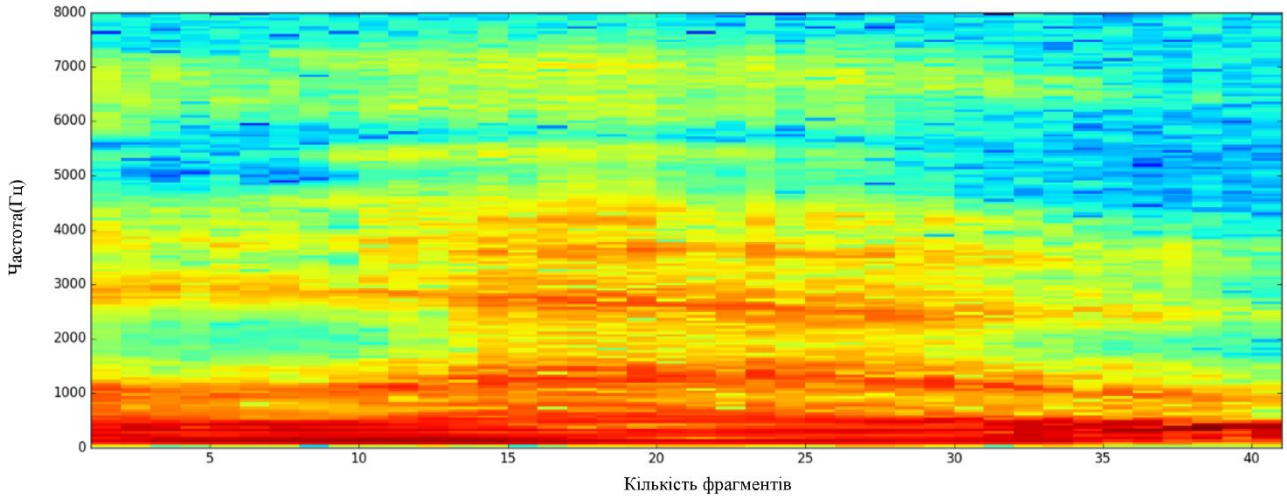


Рисунок 2.9 – Спектрограма аудіофайла

Тепер, коли отримали формат аудіо, з яким можна працювати далі, навчимо на цих даних глибоку нейромережу. Нейромережа буде отримувати аудіо-фрагменти довжиною 20 мс. Для кожного невеликого фрагмента мережа намагатиметься визначити яка саме буква була вимовлена. Модель, поточний стан якої впливає на подальші результати, зображена на рисунку 2.10 – Модель зі збереженням стану.

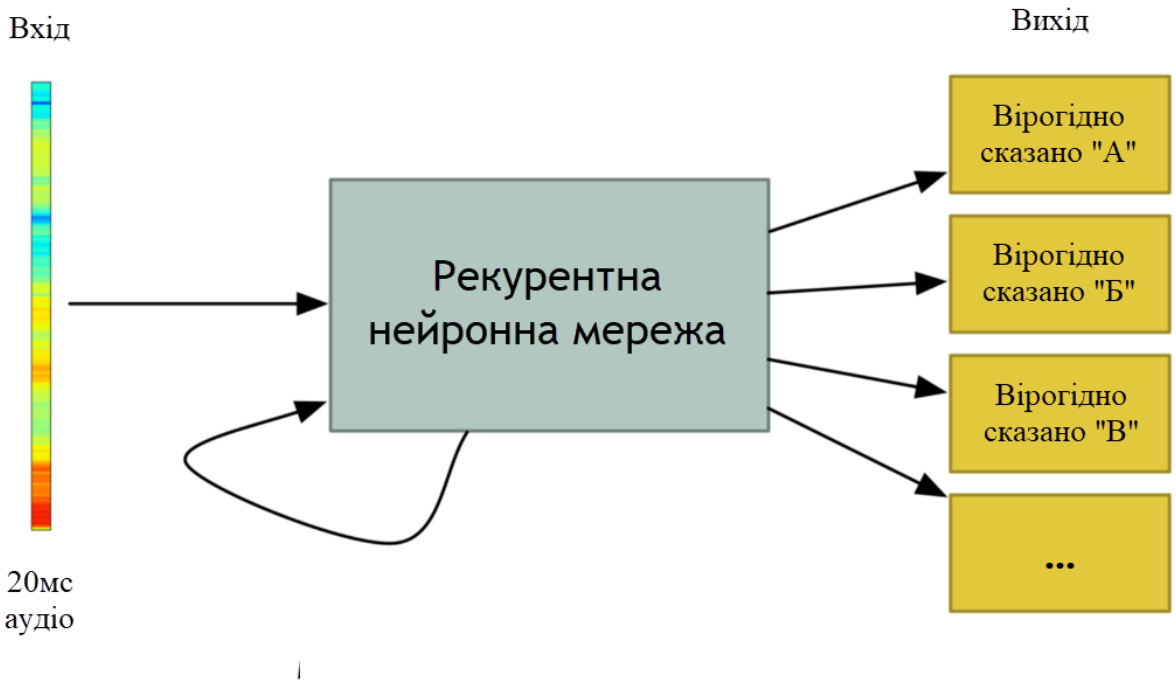


Рисунок 2.10 – Модель зі збереженням стану

Використаємо рекурентну нейронну мережу, тобто таку, яка враховує результати попередніх кроків на кожному кроці. Ми робимо це, тому що кожна буква, розпізнана мережею, має впливати на ймовірну наступну букву. Наприклад, якщо було сказано фрагмент «Прив», то скоріше за все, далі буде сказано «іт», щоб закінчити слово «Привіт». Набагато менш ймовірно, що буде сказано щось невимовне, наприклад «шхй». Отже, запам’ятавши попередні результати, нейронна мережа зможе робити більш точні передбачення в майбутньому.

Після обробки всього аудіофайлу нейронною мережою (по одному фрагменту за раз), отримуємо розклад кожного аудіо-фрагмента на літери, які, швидше за все, і вимовляються у цьому фрагменті. В результаті отримаємо майже готові слова, яких можуть повторюватись літери, міститись пробіли. Для отримання остаточного слова необхідно замінити повторювані символи одним символом і видалити пробіли.

Висновок до розділу

У розділі були розглянуті технології синтезу мовлення та розпізнавання мовлення. Були детально описані підходи до синтезу мовлення, а також розглянуто алгоритм розпізнавання мовлення, що базується на рекурентних нейронних мережах.

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ПОРІВНЯННЯ АУДІОФАЙЛІВ

3.1 Синхронізація аудіофайлів за допомогою алгоритму динамічної трансформації часової шкали

Задача, яку необхідно вирішити полягає в тому, щоб порівняти два аудіофайли різної довжини. Маємо два сигнали x та y, які можуть мати однакове звучання, але мають різну тривалість. Навіть, якщо виразити два аудіосигнали за допомогою однакової функції, підсумування їх попарних відстаней є неможливим, оскільки сигнали мають різну довжину. Ця проблема називається синхронізація аудіофайлів.

Для вирішення цієї проблеми застосовується алгоритм динамічної трансформації часової шкали (англійською DTW – Dynamic Time Wraping). Даний алгоритм застосовується для вирівнювання двох послідовностей подібного змісту, але, можливо, різної довжини.

Вимірювання відстані між двома часовими рядами необхідно для визначення їх подібності та класифікації. Одним з таких ефективних показників є метрика Евкліда. Це сума квадратів відстаней від кожної n-ї точки однієї послідовності до n-ої точки іншові. Однак використання даної метрики має суттєвий недолік: якщо два часові ряди рівні, але один з них трохи зміщений у часі (вздовж осі часу), то евклідова метрика може сигналвзувати, що ряди відрізняються один від одного. Алгоритм DTW був введений, щоб заповнити цю прогалину та забезпечити візуальне вимірювання міжрядкових інтервалів, що ігнорує як глобальні, так і локальні відхилення у часовій шкалі. [2]

Розглянемо два часових ряда – Q довжини n та C довжини m.

$$Q = q_1, q_2, \dots, q_i, \dots, q_n$$

$$C = c_1, c_2, \dots, c_j, \dots, c_m$$

Перший етап алгоритму заключається в наступному. Необхідно побудувати матрицю d порядку $n \times m$ (матрицю відстаней), в якій елемент d_{ij} є відстанню $d(q_i, c_j)$ між точками рядів q_i та c_j. У більшості випадків застосовується евклідова

відстань: $d(q_i, c_j) = (q_i - c_j)^2$ або $d(q_i, c_j) = |q_i - c_j|$. Кожний елемент *(i, j)* матриці відповідає вирівнюванню між точками q_i і c_j.

На другому етапі будуємо матрицю трансформармації, кожний елемент якої обчислюється виходячи з наступного співвідношення:

$$D_{i\,j} = d_{i\,j} + min(D_{i-1\,j}, D_{i-1\,j-1}, D_{i\,j-1})$$

Після заповнення матриці трансформації, переходимо до заключного етапу, який полягає в тому, щоб побудувати деякий оптимальний шлях трансформації і DWT відстань (вартість шляху).

W – це, шлях перетворення, представлений набором сусідніх елементів матриці, який встановлює відповідність між *Q* та *C*. Це шлях, який мінімізує загальну відстань між Це шлях, який мінімізує загальну відстань між *Q* та *C*. *k*-ий елемент шляху *W* обчислюється за наступною формулою:

$$w_k = (i, j)_k, \; d(w_k) = d(q_i, c_j) = (q_i - c_j)^2$$

Отже:

$$W = w_1, w_2, \dots, w_k, \dots, w_K; \quad max(m, n) \leq K < m + n,$$

де *K* – це довжина зазначеного шляху.

Шлях перетворення повинен відповідати наступним умовам:

– початок шляху $w_1 = (1, 1)$, та його кінець $w_K = (m, n)$. Дана умова свідчить, що шлях трансформації містить всі точки обох часових рядів.

– будь-які два сусідні елемента шляху W, $w_k = (w_i, w_j)$ і $w_{k+1} = (w_{i+1}, w_{j+1})$, є такими, що: $w_i - w_{i+1} \leq 1, w_j - w_{j-1} \leq 1$. Це обмеження гарантує, що шлях трансформації рухається крок за кроком. Тобто обидва індекси і та j можуть збільшуватися лише на 1 на кожному кроці шляху.

– будь-які два сусідні елемента шляху W, $w_k = (w_i, w_j)$ і $w_{k-1} = (w_{i-1}, w_{j-1})$, є такими, що: $w_i - w_{i-1} \geq 0, w_j - w_{j-1} \geq 0$. Це обмеження гарантує, що шлях перетворення не повернеться до минулої точки. Тобто обидва індекси *i* та *j* залишаються незмінними або зростають, але ніколи не зменшуються.

Незважаючи на те, що існує велика кількість шляхів трансформації, які задовольняють всі перераховані вище умови, потрібен лише шлях, який мінімізує

відстань DTW (вартість шляху).

Відстань DTW (вартість шляху) між двома послідовностями обчислюється на основі оптимального шляху перетворення за допомогою формули:

$$DTW(Q,C) = min\left\{\frac{\sum_{k=1}^{K} d(w_k)}{K}\right\}$$

K в знаменнику використовується для врахування того, що шляхи трансформації можуть бути різної довжини.

Просторова і тимчасова складність алгоритму – квадратична $O\ (nm)$, так як DTW алгоритм повинен вивчити кожну клітинку матриці трансформації.

3.2 Перетворення Фур’є для обробки аудіосигналів

Як відомо, звуковий сигнал у комп'ютері може бути представлений у вигляді деякого набору відліків його амплітуд, що створюються через певні проміжки часу (періоди дискретизації) та представлені деякою кількістю двійкових розрядів (розрядність виборки). Таке представлення зручне для зберігання звукового сигналу та його зворотнього перетворення в безперервний сигнал. Однак деякі операції обробки звуку в цьому поданні не завжди зручні. Це пов'язано з тим, що реальний звуковий сигнал складається з частот з визначеною амплітудою та фазою. Отже, використання операцій обробки звукового сигналу, таких як "фільтр низьких частот" або "фільтр високих частот", вимагає перетворень подань звукового сигналу як еталону в його частотному спектрі. Після цього перетворення звукового сигналу буде представлено у вигляді коефіцієнтів, що відповідають амплітудам і фазам частот, які і складають цей сигнал. Тепер, наприклад, операція "фільтр нижніх частот", що вирізає із сигналів усі частоти, які вище заданої, може просто онулити, відповідні частотам, коефіцієнти, які необхідно вирізати. Насправді, область застосування такого перетворення значно ширша: обробка растрових зображень, телекомунікації, дослідження та вимірювання сигналів, радіолокація та ін. Прикладом застосування перетворення може бути передача даних у цифровій формі по аналоговим лініями телефонної мережі (модем). Для передачі даних у цифровій формі вони починають перетворюватися в певний набір частот і передаються по лінії передачі, а потім, на приймаючій стороні, виконується

фундаментальне перетворення та відновлюються вихідні дані. Далі розглянуті фундаментальні поняття, які необхідні для розуміння роботи перетворення Фур'є. [3]

Рядом Фур'є називається нескінченна математична послідовність, яка складається з коефіцієнтів при функціях синуса та косинуса вигляду:

$$\frac{a_0}{2} + \sum_{m=1}^{\infty} \left(a_m \cos \frac{m\pi x}{l} + b_m \sin \frac{m\pi x}{l} \right)$$

Відомо, що якщоперіодична функція з періодом $2l$ на проміжку $[-l, l]$ задовільняє умови першого роду (умови Діріхле), тобто безперервна і має скінченну кількість екстремумів і точок розриву I роду, то ця функція може представлятися у вигляді суми ряду Фур'є. Для визначення коефіцієнтів ряду Фур'є використовуються наступні формули:

$$a_m = \frac{1}{l} \int_{-l}^{l} f(x) \cos \frac{m\pi x}{l} dx,$$

$$a_m = \frac{1}{l} \int_{-l}^{l} f(x) \sin \frac{m\pi x}{l} dx$$

Якщо функція, що розкладається, парна, тобто $f(-x) = f(x)$ [4], то ряд Фур'є складається лише з косинусів. Це означає, що усі коефіцієнти при синусах рівні нулю. Якщо ж функція є непарною, тобто $f(-x) = -f(x)$ [4], тоді ряд Фур'є складається лише з синусів. Це означає, що усі коефіцієнти при косинусах рівні нулю. В загальному випадку, коефіцієнти при синусах і косинусах не рівні нулю. Таким чином, будь яку періодичну функцію, яка задовільняє умови першого роду, можна розкласти в ряд Фур'є, представивши дану функцію у вигляді суми косинусів і синусів.

Уявімо, що період повторення досліджуваної функції – нескінченність. Тобто функція не є періодичною. У цьому випадку існує інтегральна формула Фур'є, яку отримують при граничному переході з ряду Фур'є періодичної функції з періодом *2l*:

$$f(x) = \frac{1}{\pi}\int_{0}^{+\infty} dz \int_{-\infty}^{+\infty} f(u)\cos z(u-x)du$$

З цією формулою пов’язані інтегральні перетворення Фур’є:

$$F(z) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{izx} f(x)dx$$

$$f(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{-izx} F(z)dz$$

З цих формул можна винести синус-перетворення Фур’є для непарних функцій:

$$f_s(z) = \sqrt{\frac{2}{\pi}}\int_{0}^{+\infty} f(x)\sin zx\, dx$$

$$f(x) = \sqrt{\frac{2}{\pi}}\int_{0}^{+\infty} f_s(z)\sin zx\, dx$$

Тепер винесемо косинус-перетворення Фур’є для парних функцій:

$$f_c(z) = \sqrt{\frac{2}{\pi}}\int_{0}^{+\infty} f(x)\cos zx\, dx$$

$$f(x) = \sqrt{\frac{2}{\pi}}\int_{0}^{+\infty} f_c(z)\sin zx\, dx$$

Отже, безперервне перетворення Фур'є дозволяє представити неперіодичну функцію у вигляді інтеграла функції, що представляє в кожній зі своїх точок коефіцієнт ряду Фур'є для неперіодичної функції.

В дискретному перетворені Фур’є (DFT – Discrete Fourier Transform) досліджувана функція є періодичною, має кінцевий період повторення та є дискретною. Для визначення амплітуд та фаз частотних складових сигналу в DFT використовується співвідношення з базисними функціями синуса та косинуса.

Спектр частот в DFT визначається з амплітуд синусів і косинусів з частотами

повторення у досліджуваній виборці від 0 до $N/2$, де N – кількість елементів вибірки. Перетворення Фур'є розкладає дискретизований сигнал з N вибірок на $N/2+1$ синусних та $N/2+1$ косинусних складових. Це призводить до наступних формул DFT:

$$ReX[k] = \sum_{i=0}^{N-1} x[i] \cos\left(\frac{2\pi ki}{N}\right)$$

$$ImX[k] = -\sum_{i=0}^{N-1} x[i] \sin\left(\frac{2\pi ki}{N}\right)$$

Ці формули описують пряме перетворення Фур'є. $ReX[k]$ – масив, у якому міститься значення косинусоідальних складових, відповідно, $ImX[k]$ – масив синусоідальних складових.

По наведеним вище формулам відбувається розкладання досліджуваного сигналу в його спектр.

Після розкладу сигналу на синусоідальні та косинусоідальні складові можна отримати значеннння амплітуд та фаз частотних складових сигналу. Для переводу пари відповідних коефіцієнтів при синусі і косинусі в амплітуду і фазу частотної складової використаємо формули:

$$MagX[k] = (ReX[k]^2 + ImX[k]^2)^{1/2}$$

$$PhaseX[k] = arctan\left(\frac{ImX[k]}{ReX[k]}\right)$$

Отримані значення амплітуди і фази називають полярним представленням. Значення коефіцієнтів при синусах і косинусах характеризують прямокутне представлення. Значення фази сигналу несе у собі інформацію про форму і зміни сигналу.

Швидке перетворення Фур'є (FFT – Fast Fourier Transform) – це алгоритм, який часто використовується для обробки цифрових сигналів при трансформації дискретних даних з часового представлення у частотний діапазон. Швидке перетворення Фур'є є однією із реалізації перетворення Фур'є. Алгоритм FFT обчислює дискретне перетворення Фур'є (DFT – Discrete Fourier Transform) або

зворотнє дискретне перетворення (IDFT – Inverse Discrete Fourier Transform). Оскільки обчислення DFT є надто повільним, часто використовують FFT, який швидко обчислює такі перетворення, розкладаючи матрицю DFT на добуток розсіяних (переважно нульових) факторів. Таким чином алгоритму FFT вдається зменшити складність DFT з $O(n^2)$ до $O(n \log n)$, де n – розмір даних. Різниця в швидкості може бути дуже значною, особливо для великих наборів даних, де N може складати тисячі або мільйони.

3.3 Середньоквадратична помилка

Після виконання перетворення Фур'є отримано набір характерних частот двох аудіосигналів. Для визначення відносної різниці між ними використовується метод визначення середньоквадратичної помилки.

Середньоквадратична помилка визначається наступною формулою:

$$E = \frac{1}{n}\sum_{i=1}^{n}(MagA_i - MagB_i)^2,$$

де $MagA_i$ – це амплітуда частоти i в сигналі A, а $MagB_i$ – це амплітуда частоти i в сигналі B.

При визначені середньоквадратичної помилки, різницею в фазових складових нехтується, оскільки вони не несуть корисної інформації.

Висновки до розділу

У розділі були розглянуте математичне забезпечення для реалізації алгоритму порівняння аудіофайлів. Було розглянуто алгоритм динамічної трансформації часової шкали для синхронізації аудіофайлів, перетворення Фур'є для обробки аудіосигналів. Також, було описано метод обчислення середньоквадратичної помилки для визначення відносної різниці між аудіофайлами.

Висновки до розділу

У даному розділі було розглянуто метод порівняння двох аудіофайлів, що може бути використаний з метою визначення правильності вимови іншомовних

слів. Було розглянуто алгоритм динамічної трансформації часової шкали для синхронізаціїї аудіофайлів, дискретне перетворення Фур’є та швидке перетворення Фур’є для представлення аудіосигналів у дискретному вигляді. Для відображення результату порівняння двох аудіофайлів було обрано середньоквадратичну помилку.

4 ОПИС АРХІТЕКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПОСТАНОВКИ ВИМОВИ

4.1 Архітектура програмного забезпечення

Мова програмування Java була обрана для розробки веб-застосунку через її властивості:

– простота – Java має стислі, тісно пов’язані та прості у вивченні мовні інструменти;

– безпека – Java - це надійний інструмент для створення веб-застосунківі;

– портативність – додатки Java можна запускати в будь-якому середовищі, де є виконуюча система Java;

– об’єктно-орієнтована природа – Java втілює сучасну філософію об’єктно-орієнтованого програмування;

– надійність – Java зменшує ймовірність помилок у програмах за рахунок жорсткого набору змінних та реалізації відповідних елементів керування під час виконання;

– багатопотоковість – Java забезпечує інтегровану підтримку багатопотокового програмування;

– архітектурна незалежність – Java не прив’язана до певного типу обчислювальної машини або архітектури операційної системи;

– інтерпретабельність – Java забезпечує байт-код, який забезпечує незалежність від платформи;

– висока продуктивність – байт-код Java оптимізований для підвищення продуктивності;

– розподіленність – Java розроблена з урахуванням її застосування в розподіленому Інтернет-середовищі;

– динамічність – програми Java містять значну частину інформації, яка використовується під час виконання для перевірки та надання доступу до об'єктів.[5]

Застосування побудовано на основі архітектурного шаблону MVC. Схема

шаблону MVC наведена на рисунку 4.1 – Структура шаблону MVC.

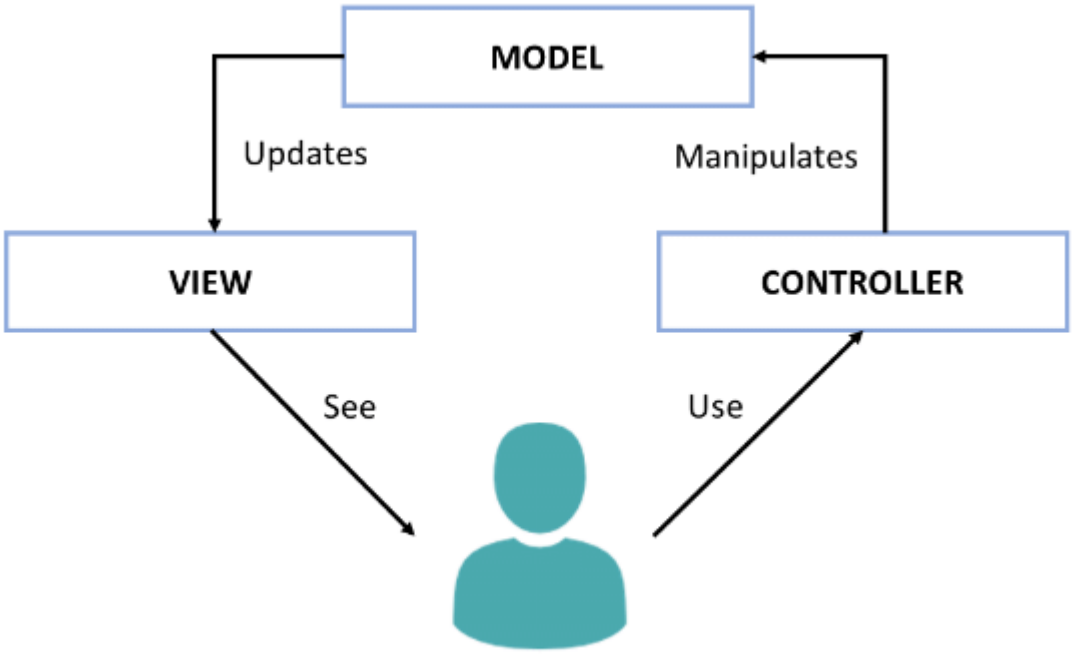


Рисунок 4.1 – Структура шаблону MVC

MVC – це шаблон дизайну програмного забезпечення, який зазвичай використовується для розробки інтерфейсу користувача, який розділяє відповідну логіку програми на три взаємопов'язані елементи. Це робиться для відокремлення внутрішнього подання інформації від способів подання інформації та прийняття інформації користувачем. [6]

Модель (Model) – центральний компонент шаблону MVC. Це динамічна структура даних програмного забезпечення, яка незалежна від інтерфейсу користувача. Цей компонент безпосередньо управляє даними, логікою та правилами програмного забезепечення. Модель відповідає за управління даними програми. Компонент отримує ввід користувача від контролера.

Представлення (View) – компонент шаблону MVC, який відповідає за представлення об’єктів у програмному забезпеченні, реагуючи на зміни моделі. Представлення означає презентацію моделі в певному форматі.

Контролер (Controller) – компонент шаблону MVC, який оновлює як модель, так і представлення. Він отримує ввід користувача та перетворює його в команди для моделі чи представлення. Контролер реагує на ввід користувача і виконує

взаємодію з об'єктами моделі даних. Контролер отримує вхідні дані, додатково перевіряє їх, а потім передає вхідні дані для моделі.

Spring Framework – універсальний фреймворк, що забезпечує повну модель розробки та конфігурації сучасних бізнес-застосунків, написаних на мові програмування Java. Ключовим елементом Spring є підтримка інфраструктури на рівні програми, тобто розробники можуть зосередитися на бізнес-логіці програми без додаткових налаштувань в залежності від середовища виконання [7].

Основними можливостями Spring є:

– введення залежностей;

– аспектно-орієнтоване програмування, включаючи декларативне управління транзакціями;

– створення додатків Spring MVC;

– Spring можна розглядати як сукупність інших, менших фреймворків або фреймворків у фреймворку. Більшість цих фреймворків можуть працювати самостійно, але при спільному використанні ці фреймворки надають більше функціональних можливостей.

– контейнер інверсія управління: конфігурація компонентів програми та управління життєвим циклом об'єктів Java;

– аспектно-орієнтована система програмування. Цей фреймворк допомагає працювати з функціоналом, який неможливо без втрат включити в об’єктно-орієнтовані можливості програмування на Java;

– фреймворк доступу до даних. Цей фреймворк працює із системами управління базами даних на платформі Java, використовуючи інструменти JDBC та ORM, і пропонує рішення проблем, які у середовищі Java повторюються у великій кількості.

– фреймворк керування операціями: координація різних API управління транзакціями та інструментів конфігурації управління транзакціями для об'єктів Java;

– фреймворк MVC. Даний фреймворк підтримує архітектурний шаблон MVC та реалізує всі основні функції ядра Spring. Детальне зображення схеми

взаємодії компонентів Spring MVC представлена на рисунку 2.6 - Схема взаємодії компонентів Spring MVC.

– фреймворк аутентифікації і авторизації: надає інструменти для реалізації та налаштування процесів автентифікації та авторизації, які підтримують багато стандартних протоколів, інструментів та практик через дочірній проект Spring Security. [7]

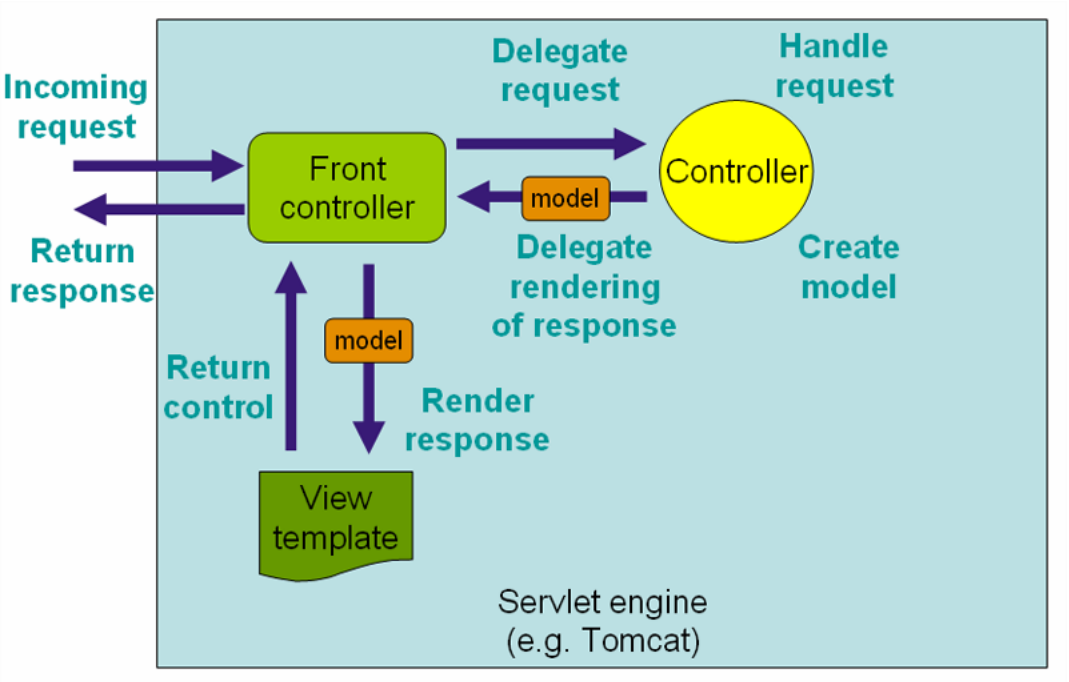


Рисунок 4.2 – Схема взаємодії Spring MVC

Hibernate - це Java бібліотека, що розроблена для вирішення проблем об'єктного реляційного відображення (ORM) [8].

Мета Hibernate - полегшити розробнику роботу зі значним обсягом низькорівневого програмування при роботі з об’єктно-орієнтованими додатками в реляційних базах даних. Зазвичай розробники використовують Hibernate не тільки для полегшення процесу проектування системи класів і таблиць з самого початку, але й для роботи з існуючою базою даних [8].

Hibernate не тільки вирішує взаємозв'язок класів Java з таблицями в базі даних (а типи даних Java - з типами даних SQL), але також дозволяє автоматизувати процес створення та оновлення таблиць, запитів до бази даних та обробки результатів з цих запитів із використанням спеціальних фондів. Крім того, цей фреймворк допомагає значно скоротити час розробки додатків, який витрачається

на написання програмістом коду SQL та JDBC. Hibernate автоматично генерує запити SQL і звільняє програміста від ручного адміністрування результуючого набору даних та перетворення об'єктів, що максимально спрощує передачу додатків до будь-якої бази даних SQL [8].

Як базу даних вибрано PostgreSQL. PostgreSQL - це вільно розподілена об'єктна система управління реляційними базами даних (ORDBMS), найбільш розроблена серед відкритих СУБД у світі і є реальною альтернативою комерційним базам даних [9]. Підтримується на всіх сучасних системах Unix, включаючи найпоширеніші, таких як Linux, FreeBSD, NetBSD, OpenBSD, SunOS, Solaris, DUX, а також Mac OS X. Починаючи з версії 8.X PostgreSQL працює в "рідному" режимі під MS Windows .

4.2 Конструювання програмного забезпечення

Як зазначалося вище, архітектура розробленої системи базується на моделі MVC. Репозиторії Spring Data знаходяться на рівні доступу до даних. Вони використовуються службами на рівні бізнес-логіки, які в свою чергу використовуються контролерами даних, які обробляють запити користувача та відображають їх у вигляді веб-сторінок.

Об'єкти, з якими ми працюємо, - це сутності, які мають безпосереднє представлення в базі даних. Структурна схема бази даних представлена у графічному матеріалі.

Діаграма класів моделей, що відповідає сутностям в базі даних наведена на рисунку 4.4 – Схема структурна класів моделей.

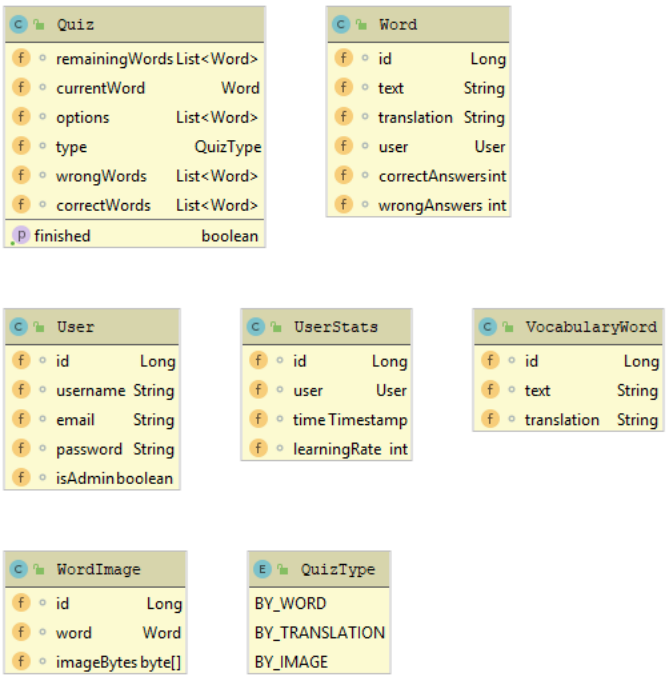


Рисунок 4.3 – Схема структурна класів моделей

Бізнес логіка повністю реалізована в класах сервісів. Структурна схема діаграми інтерфейсів та їх реалізації сервісів зображена на рисунку 4.5 – схема структурна класів сервісів.

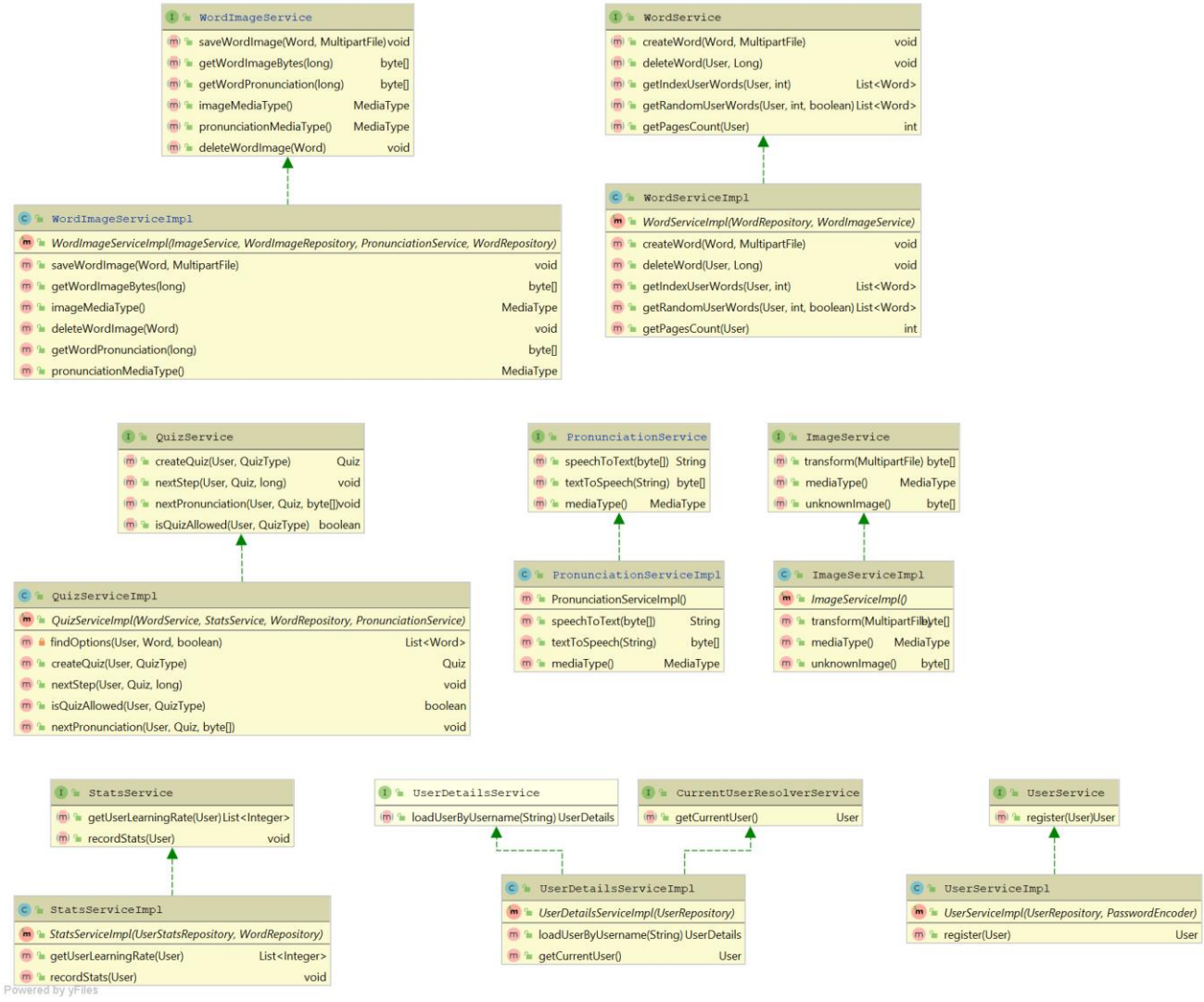


Рисунок 4.4 – Схема структурна діаграми класів Сервісів

4.3 Програмні засоби для реалізації розробленого методу визначення правильності вимови

Для реалізації розробленого методу використано бібліотеку librosa.

librosa – це Python бібліотека для аналізу музики та аудіо. Вона надає будівельні блоки, необхідні для створення систем пошуку аудіоінформації. Хоча сама бібліотека і призначена для пошуку аудіоінформації, вона може бути використана в розроблюваному алгоритмі порівняння аудіофайлів, оскільки містить необхідні методи для реалізації алгоритму порівняння аудіофайлів.

4.4 Способи взаємодії між процесами

Оскільки, серверна частина застосунку розроблена на мові програмування Java, а алгоритм порівняння аудіофайлів розробленийна мові програмування

Python, необхідно реалізувати взаємодію між процесами (inter-process communication – IPC).

Різні підходи до IPC були пристосовані до різних вимог до програмного забезпечення, таких як продуктивність, модульність та системні обставини, такі як пропускна здатність мережі та затримка. Основні підходи описані у таблиці 4.1

Таблиця 4.1 – Підходи до IPC

Метод	Опис
Файл	Запис, що зберігається на диску, або запис, синтезований на вимогу файловим сервером, до якого можна отримати доступ за допомогою декількох процесів.
Файл зв'язку	Унікальна форма IPC наприкінці 1960-х, яка найбільше нагадує протокол 9P плану 9
Сигнал	Системне повідомлення, надіслане з одного процесу в інший, зазвичай не використовується для передачі даних, а замість цього використовується для віддаленого керування процесом, що складається з партнерів.
Сокет	Дані надсилаються через мережевий інтерфейс або до іншого процесу на тому самому комп'ютері, або до іншого комп'ютера в мережі.
Unix сокет	Подібно до інтернет-сокета, але все спілкування відбувається всередині ядра. Сокети домену використовують файлову систему як свій адресний простір. Процеси посилаються на сокет домену як на анод, і кілька процесів можуть взаємодіяти з одним сокетом
Черга повідомлень	Потік даних, схожий на сокет, але який зазвичай зберігає межі повідомлень. Зазвичай реалізовані операційною системою, вони дозволяють безлічі процесів читати та записувати в чергу повідомлень, не будучи безпосередньо підключеними один до одного.

Продовження таблиці 4.1

Анонімний ріре	Односпрямований канал даних, що використовує стандартний вхід і вихід. Дані, записані на кінець запису конвеєра, буферизуються операційною системою, доки вони не будуть зчитуватися з кінця конвеєра конвеєра. Двосторонній зв’язок між процесами можна досягти, використовуючи дві труби в протилежних «напрямках».
Іменований ріре	Ріре, який обробляється як файл. Замість того, щоб використовувати стандартні вхідні та вихідні дані, як для анонімного, процеси записують і читають із іменованого, ніби це звичайний файл.
Спільна пам’ять	Кілька процесів отримують доступ до одного блоку пам'яті, що створює спільний буфер для взаємодії процесів між собою.
Передача повідомлення	Дозволяє декільком програмам спілкуватися за допомогою черг повідомлень та / або каналів, якими не керує ОС. Зазвичай використовується в моделях паралельності.
Файл із трансляцією пам'яті	Файл, зіставлений з оперативною пам'яттю, може бути змінений шляхом безпосередньої зміни адрес пам'яті замість виведення в потік. Це надає ті ж переваги, що і стандартний файл.

Для реалізації взаємодії між процесами було обрано підхід з використанням файлів та зчитування виводу дочірнього процесу (Ріре).

Висновки до розділу

У цьому розділі були детально розглянуті технології, засоби та архітектурні шаблони, що використовуються в розроблюваній системі. Була наведена та описана схема бази даних, а також структурна схема діаграми інтерфейсів і структурна схема діаграми класів, що реалізують інтерфейси. Також були розглянуті способи взаємодії між процесами.

5 РОЗРОБКА СТАРТАП ПРОЕКТУ

5.1 Опис ідеї проекту

В межах даного підпункту послідовно проаналізовано та подану у вигляді таблиць:

– зміст ідеї – система розпізнавання мовлення у системі для вивчення іноземних мов;

– можливі напрямки застосування;

– основні вигоди, що отримує користувач;

– чим відрізняється розроблена система від існуючих аналогів.

У таблиці 5.1 описана ідея стартап-проекту, а саме розкритий зміст ідеї, наведені напрямки застосування та сформульовані основні вигоди, що отримує користувач.

Таблиця 5.1 – Опис ідеї стартап проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Розробка програмного застосунку для вивчення іноземних мов, який дозволятиме не лише вивчати слова, а у постановці вимови за допомогою використання технології розпізнавання мовлення.	1. Вивчення іноземних слів	Для користувача вирішується одна із найкладніших проблем при вивченні іноземних мов – поповнення словникового запасу
	2. Перевірка вимови, як один із типів завдань	Користувач може тренувати вимову слів та покращувати вимову окремих слів, які користувачеві складно вивчити

Продовження таблиці 5.1

	3. Прослуховування правильної вимови	Користувач має можливість не просто повторювати слова, а і перевірити як правильно вимовляється слово
	4. Аналіз вимови та надання результатів	Користувач зможе бачити на транскрипції, яку саме частину мови він вимовив неправильно та проводити роботу над помилками.

Також, проведено аналіз потенційних техніко-економічних переваг ідеї розроблюваного стартап-проекту. В таблиці 5.2 наведені результати досліджень, а саме:

– визначено перелік техніко-економічних властивостей та характеристик ідеї;

– визначено попереднє коло конкурентів (проектів-конкурентів) або товарів-замінників чи товарів-аналогів, що вже існують на ринку, та проведено збір інформації щодо значень техніко-економічних показників для ідеї власного проекту та проектів-конкурентів відповідно до визначеного вище переліку;

– проведено порівняльний аналіз показників: для власної ідеї визначено показники, що мають:

1) гірші значення (W, слабкі);
2) аналогічні (N, нейтральні) значення;
3) кращі значення (S, сильні).

Для порівняння були обрані два найбільш схожі конкуренти, а саме Uter та Elsa.

Таблиця 5.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	Концепції конкурентів			Слабкі (W), нейтральні (N) та сильні (S) сторони		
		Мій проект	Uter	Elsa	W	N	S
1.	Можливість додавати власні слова	Так	Ні, лише існуючі в системі слова	Ні, набір слів обмежений – 2000			+
2.	Різні типи завдань, які зосереджені не на постановці вимови	Так	Ні, лише постановка вимови	Ні, лише постановка вимови			+
3.	Завдання постановки вимови	Так	Так, але користувач обмежений словами	Так, але користувач обмежений словами			+
4.	Можливість прослухати правильне звучання слова	Так	Так	Так		+	
5.	Можливість переглянути в якій частині слова вимова була неправильною	Так	Ні	Так			+

Продовження таблиці 5.2

6.	Підрахунок прогресу вивчення слів	Так	Ні	Так			+

Провівши аналіз та порівняння з попереднім колом проектів, було визначено, що у розроблюваного застосунку висока конкурентоспроможність, оскільки навіть найближчі аналоги не мають усіх функцій мого проекту.

5.2 Технологічний аудит ідеї проекту

В межах даного підрозділу було проведено аудит технологій, за допомогою яких можна реалізувати ідею проекту.

Технологічу здійсненність ідеї проекту було проведено за допомогою аналізу таких складових:

– технологія, за якою буде виготовлено товар згідно ідеї проекту;
– існування технології: технологія вже існує чи її потрібно розробити;
– доступність технології.

Технологічну здійсненність ідеї проекту наведено у таблиці 5.3.

Таблиця 5.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технологія реалізації	Наявність технології	Доступність технології
1.	Розпізнавання мовлення користувача	Google Speech-To-Text	Наявна	Безкоштовна до 60 хвилин аудіо, далі $0.006/15 секунд
2.	Озвучення будь-якого слова зі словнику користувача	Google Text-to-Speech	Наявна	Платна, $4.00 USD / 1 мільйон символів

Продовження таблиці 5.3

3.	Порівняння аудіофайлів	Математичне забезпечення	Наявна	Безкоштовна
4.	Позначення слова, яке було неправильно вимовлене			

За результатами аналізу таблиці, зроблено висновок, що технологічна реалізація проекту можлива і визначено технологічний шлях, яким доцільно реалізувати ідею проекту.

5.3 Аналіз ринкових можливостей запуску стартап-проекту

В межах даного підпункту було визначено ринкові можливості, які можна використати під час ринкового впровадження проекту, та ринкоі загрози, які можуть перешкодити реалізації проекту, що дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

Перш за все, у таблиці 5.4 було проведено аналіз попиту, а саме наявність попиту, обсяг, динаміка розвитку ринку.

Таблиця 5.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1.	Кількість головних гравців, од	2
2.	Загальний обсяг продаж, грн/ум.од	1000 грн/ум.од міс
3.	Динаміка ринку	Зростає
4.	Наявність обмежень для входу	Немає
5.	Специфічні вимоги до стандартизації та сертифікації	Немає
6.	Середня норма рентабельності в галузі, %	Дані відсутні

За результатами аналізу таблиці зроблено висновок що, за попереднім

оцінюванням, ринок є привабливим для входження.

В таблиці 5.5 визначено потенційні групи клієнтів, їх характеристики, та сформовано орієнтовний перелік вимог до товару для кожної групи.

Таблиця 5.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1.	Можливість вивчати іноземні слова без першкод	Фізичні особи, які хочуть вивчати іноземні мови, та для яких запам’ятовування слів є складним завданням, підприємства, на яких працівникам необхідно покращувати знання іноземних мов	Підприємства готові надавати корпоративну знижку клієнтам, що впливає на кількість користувачів	Наявність веб інтерфейсу, мобільного додатку, зручність та простота у використанні
2.	Постановка вимови без залучення третіх осіб	Фізичні особи	Немає, або незначні	Наявність веб інтерфейсу, мобільного додатку, зручність та простота у використанні.

Після визначення потенційних груп клієнтів проводено аналіз ринкового

середовища та складено таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають.

Фактори, що сприяють ринковому впровадженню проекту наведені у таблиці 5.6.

Таблиця 5.6 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1.	Наявність функціоналу, якого немає у конкурентів	Немає конкурентів, у яких або є інші завдання, окрім постановки вимови, або є типи завдань, націлені саме на вивчення слів, але без постановки вимови	У рекламній кампанії акцентувати увагу на переваги продукту над конкурентами. Розширення набору існуючих функцій.
1.	Зниження довіри до конкурента	Через ненадійність ПЗ конкурента його клієнтська база зменшується	У рекламній компанії акцентувати увагу на переваги продукту над конкурентом.
2.	Зростання цільової аудиторії продукту	Через розширення можливостей для подорожей та роботи за кородоном, зростає необхідність знання іноземних мов	Вихід на глобальний ринок

Фактори, що перешкоджають ринковому впровадженню проекту наведені у

таблиці 5.7.

Таблиця 5.7 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Конкуренція	Поява нових конкурентів на ринку	Зміна цінової політики. Розширення функціоналу. Розширення рекламної кампанії.
3.	Непрогнозована велика популярність системи	Через непрогнозоовано великий попит на систему не вистачає обчислювальних можливостей для її нормальної роботи	Перехід на хмарний хостинг з підтримкою динамічного горизонтального масштабування; Збільшення вартості експлуатації системи.

Надалі проведено аналіз пропозиції: визначено загальні риси конкуренції на ринку. Ступеневий аналіз конкуренції на ринку наведено у таблиці 5.8

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції Чиста	Існують додатки з подібним функціоналом	Інтерфейс має бути максимально зручним та привабливим, завдання – унікальні, а ціна - низькою

Продовження таблиці 5.8

2. За рівнем конкурентної боротьби Національний	Направленість додатку виражається особливостями, притаманними людям конкретної країни	Орієнтованість на потреби українця
3. За галузевою ознакою Внутрішньогалузева	Товари задовольняють одну потребу, мають відмінності в ціні, функціях, інтерфейсі	Низька ціна, бонуси, знижки
4. Конкуренція за видами товарів: Товарно-видова	Існують застосування зі схожим функціоналом	Створення унікальних можливостей для користувача
5. За характером Нецінова	Більшість додатків безкоштовні	Мають бути створені унікальні функції
6. За інтенсивністю Не марочна	Більшість компаній ноу-нейми	Великою перевагою стане співпраця з школами іноземних мов

Після аналізу конкуренції в таблиці 5.9 проведено більш детальний аналіз умов конкуренції в галузі.

Таблиця 5.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники

Продовження таблиці 5.9

	Додатки – словники, додатки – інтерактивні завдання, додатки для постановки вимови	Додатки-аналоги з підлаштуванням під користувача	Мережа Інтернет	фінансовий	Замінники простіші для розуміння, дешевші
Висновки:	Конкуренція висока	Поки не анонсовані, є шанс бути першим таким додатком	Треба відповідати стандартам веб-сервісів	Не диктують, розраховано на масовий ринок	Надати безкоштовний пробний період

За результатами аналізу конкуренції зроблено висновок, що ринок є відкритим для входу та потребує товару, який буде якісно кращим за наявні рішення та надавати більше функціональності, ніж у товарів-конкурентів.

На основі аналізу конкуренції, а також із урахуванням характеристик ідеї проекту, вимог споживачів до товару та факторів маркетингового середовища визначано та обґрунтовано перелік факторів конкурентоспроможності у таблиці 5.10.

Таблиця 5.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)

Продовження таблиці 5.10

1	Новаторство	Не існує товарів конкурентів, які пропонують користувачам комплексний підхід до вивчення слів. Конкуренти зосереджуються або на вивченні слів за допомогою різних завдань або на постановці вимови.
2	Наявність постійних оновлень	Постійне поповнення кількості та різноманітності завдань робить досвід користування продукту більш яскравим і привабливим
3	Простота інтерфейсу	Програмний продукт має слугувати помічником а не додатковою перепоною на шляху вивчення інозменої мови
4	Швидкість обробки інформації	Користувач хоче отримати додавати слова до словника швидко. Користувач хоче, щоб завдання генерувалися швидко.

За визначеними факторами конкурентоспроможності проводено аналіз сильних та слабких сторін стартап-проекту у таблиці 5.11

Таблиця 5.11 – Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з «EasyLearn»						
			–3	*–2*	*–1*	*0*	*+1*	*+2*	*+3*
1	Новаторство	20	+						
2	Наявність постійних оновлень	17			+				
3	Простота інтерфейсу	20				+			
4	Швидкість обробки інформації	18				+			

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу на основі виділених ринкових загроз та можливостей, та

сильних і слабких сторін (табл. 11).

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення. Наприклад: зниження доходів потенційних споживачів – фактор загрози, на основі якого можна зробити прогноз щодо посилення значущості цінового фактору при виборі товару та відповідно, – цінової конкуренції (а це вже – ринкова загроза).

У таблиці 5.12 наведено SWOT-аналіз стартап-проекту.

Таблиця 5.12 – SWOT- аналіз стартап-проекту

Сильні сторони: унікальний функціонал, привабливий простий інтерфейс, безкоштовні оновлення	Слабкі сторони: наявність постійних оновлень
Можливості: приваблива ціна, унікальні канали збуту	Загрози: конкуренція, відсутність інформації про продукт

На основі SWOT-аналізу були розроблені альтернативи ринкової поведінки для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок

Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів у таблиці 5.13

Таблиця 5.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації

Продовження таблиці 5.13

1	Активна реклама (соцмережі, школи іноземних мов, реклама в додатках)	Висока, за рахунок того що товар буде на слуху	2 міс на рекламну компанію
2	Система бонусів і безкоштовного тріал-періоду	Середня, користувачі можуть звикнути до функціоналу і почати за нього платити	1 рік
3	Співпраця з школами іноземних мов	Середня, поповнення користувачів за рахунок клієнтів шкіл іноземних мов	1 рік

5.4 Розроблення ринкової стратегії

Першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів. Опис цільових груп потенційних споживачів наведено у таблиці 5.14.

Таблиця 5.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент

Продовження таблиці 5.14

1	Фізичні особи	Готові	Високий	Середня, є програми-аналоги, їх відносно небагато, але функціоналу менше	Просто, попит високий, аналоги не мають функціоналу, який пропонує даний продукт
3	Школи іноземних мов	Готові розглянути як варіант покращень ефективності курсів і як наслідок збільшення клієнтів	Середній	Середня, є програми-аналоги	Середня складність, оскільки треба переконати в корисності
4	Підприємства, на яких працівникам необхідно покращувати знання іноземних мов	Не готові, бояться ризикувати і закупати корпоративну версію програми	Малий	Низька, попит не великий	Складно через неготовність до сприйняття
Які цільові групи обрано: Фізичні особи, школи іноземних мов					

За результатами аналізу потенційних груп споживачів (сегментів)було обрано цільові групи, для яких планується пропозиція товару, та визначено стратегію охоплення ринку за наступною логікою :

– якщо фокус базується на одному сегменті – оберемо стратегію концентрованого маркетингу;

– якщо співпраця планується із кількома сегментами, розробляючи для них окремо програми ринкового впливу – використаємо стратегію диференційованого маркетингу;

– якщо планується співпраця зі всім ринком, пропонуючи стандартизовану програму (включно із характеристиками товару/послуги) – використаємо масовий маркетинг.

Для роботи в обраних сегментах ринку було сформувано базову стратегію розвитку (таблиця 5.15) та сформовано базову стратегію конкурентної поведінки (таблиця 5.16).

Таблиця 5.15 – Визначення базової стратегії розвитку

№ п / п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Орієнтація на користувачів з базовим знанням англійської	Ставка на цілісний підхід до вивчення іншомовних слів: вивчення слів та тренування їх вимови та використаних методик вивчення, що робить продукт унікальним	Оптимальне співвідношення ціна / функціонал Простота у використанні продукту	Стратегія диференціації – товар відмінний від конкурентів за своїм функціоналом (персоналізація під користувача)

Таблиця 5.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Так	Забирати існуючих споживачів у конкурентів за рахунок унікальності, а також пошук нових споживачів, які ще не пробували користуватись подібним продуктом	Не буде	Стратегія лідера: наступальна (охопити максимально ринок, забрати користувачів у конкурентів)

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту, а також в залежності від обраної базової стратегії розвитку та стратегії конкурентної поведінки розробляно стратегію позиціонування, що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати проект. Стратегія позиціонування описана у таблиці 5.17.

Таблиця 5.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Швидкість обробки даних	Стратегія диференціації	Висока швидкість за низьку ціну	Висока якість продукту, задовільна робота на усіх девайсах, на яких є браузер та Інтернет
2	Зручний інтерфейс	Стратегія диференціації	Правило «3х кліків», великі кнопки, мало текстового вводу	Простота використання для будь-кого, не треба розбиратись як працює
3	Аналіз вимови	Стратегія диференціації	Використання технології Text-to-Speech та порівняння аудіофайлів	Тренування вимови
4	Наявність завдань для вивчення слів	Стратегія диференціації	Генерація завдань з використанням усіх методик ефективного вивчення слів	Швидке та ефективне вивчення слів

Результатом виконання підрозділу є узгоджена система рішень щодо ринкової поведінки стартап-компанії, яка визначає напрями роботи стартап-компанії на ринку.

5.5 Розроблення маркетингової програми стартап-проекту

Першим кроком є формування маркетингової концепції товару, який отримає споживач. Для цього у таблиці 5.18 підсумовано результати попереднього аналізу конкурентоспроможності товару.

Таблиця 5.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Додати слово	Можливість додати власне слово або фразу, а не обрати з уже сформованого переліку слів	Можливість користувача додавати власні слова
2	Вивчати слова за допомогою завдань	В основі завдань лежать найефективніші методики вивчення слів	Використання різних методик вивчення
3	Тренування вимови	Користувач має можливість перевірити та покращити свою вимову окремих слів та фраз, а також прослухати правильну вимову слів	Точний алгоритм визначення правильності вимови, можливість покращити вимову будь-яких слів, а не лише доступних в системі.

Надалі розроблено трирівневу маркетингову модель товару: уточнено ідею продукту та/або послуги, його фізичні складові, особливості процесу його надання. Трирівнева маркетингова модель представлена у таблиці 5.19.

Таблиця 5.19 – Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Веб-застосування та ненативний додаток на смартфон, що надає користувачеві можливість вивчення слів за допомогою різних завдань, в основі яких лежать найефективніші методики вивчення слів. Один із типів завдань націлений на постановку вимови.</td></tr>
<tr><td rowspan="11">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>1. Вартість обслуговування
2. Знижки
3. Безвідмовність
4. Собівартість
5. Зручність інтерфейсу
6. Стильний дизайн
7. Вимогливість до ресурсів девайсу</td><td>М
Нм
М
М
М
Нм
М</td><td>Вр
Вр
Тх
Тх
Е
Ор
Тх</td></tr>
<tr><td colspan="3">Якість: стандарти ISO для ПО, SOAP, WSDL, UDDI</td></tr>
<tr><td colspan="3">Пакування: веб-застосування</td></tr>
<tr><td colspan="3">Марка: ELInc, EasyLearn</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">До продажу – знижка на підписку на 3 і більше місяці 10%</td></tr>
<tr><td colspan="3">Після продажу – безкоштовні оновлення, щомісячні бонуси</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання: захист ідей алгоритмів.</td></tr>
</table>

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (остаточне визначення ціни

відбувається під час фінансово-економічного аналізу проекту), яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів. Аналіз проводиться експертним методом. Визначення меж встановлення ціни наведено у таблиці 5.20.

Таблиця 5.20 – Визначення меж встановлення ціни

№ п/п	Функціонал	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	Ведення списку слів	Безкоштовні	Безкоштовні	10 – 15 тис грн	Нижня: Безкоштовні Верхня: Безкоштовні
2	Завдання на вивчення слів	-	Безкоштовні або 1-3 долари(28-84грн)	10 – 15 тис грн	Нижня: Безкоштовні Верхня: 3 долари (84грн)
3	Постановка вимови	-	Безкоштовні, але мають обмеження або 3-5 доларів(84 - 140)	10 – 15 тис грн	Нижня: 2 долари (56грн) Верхня: 5 доларів (140грн)
3	Загалом	-	-	10 – 15 тис грн	Нижня: Безкоштовні Верхня: 10 доларів (250грн)

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення:

– проводити збут власними силами або залучати сторонніх посередників (власна або залучена система збуту);

– вибір та обґрунтування оптимальної глибини каналу збуту;

– вибір та обґрунтування виду посередників.

Система збуту описана у таблиці 5.21 - Формування системи збуту.

Таблиця 5.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Пробують безкоштовні версії товару протягом 7 днів, якщо сподобалось оплата щомісячної підписки	Реклама, місце в рейтингу програм	Канал одного рівня	Через веб-застосування
2	Готові платити за розширення можливостей невелику ціну	Реклама, підтримка функції перевірки вимови, акції та бонуси за користування	Канал дворівневий	Через веб-застосування

Останньою складовою маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів. Концепція маркетингових комунікацій наведена у таблиці 5.22.

Таблиця 5.22 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Очікують максимальну ефективність вивчення слів, а також постановки вимови слів	Соцмережі, пошта, сповіщення на смартфон	Комплексний підхід	Повідомлення має переконати споживача що користування додатком полегшить процес вивчення слів, збільшить ефективність та швидкість вивчення іноземних слів, а також допоможе у нелегкій задачі постановки вимови	Learn it easy!

Висновок до розділу

У даному було сформульовано та описано ідею стартап-проекту з аналізом потенційних техніко-економічних переваг цієї ідеї. Було проведено аудит технології, за допомогою якої можна реалізувати ідею проекту та визначено технологічний шлях, який є доцільним для реалізації даної ідеї.

Також було визначено ринкові можливості, які можна використати під час ринкового впровадження проекту, та ринкові загрози, які можуть перешкодити реалізації проекту, було проаналізовано стан ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

В результаті проведених досліджень було виявлено, що ідея стартап-проекту технічно може бути реалізованою а також має високу конкурентоспроможність. Також було розроблено ринкову стратегію, а саме було визначено стратегії охоплення ринку, сформовано базову стратегію розвиткуб визначено стратегію позиціонування.

Фінальним етапом даного розділу було розроблення маркетингової програми стартап-проекту.

ВИСНОВОК

В даній магістерській дисертації описано та розроблену архітектуру програмного забезпечення інтелектуальної системи розпізнавання мовлення для вивчення іншомовних слів. В ході роботи були сформовані та реалізовані задачі, для розробки даної системи. Були проаналізовані вже відомі технічні рішення та наявні аналоги, виділені їх переваги та недоліки.

Детально були розглянуті технології розпізнавання та синтезу мовлення. Для реалізації порівняння вимови було розроблено алгоритм порівняння аудіофайлів та розглянуті математичні методи для реалізації даного алгоритму.

В результаті проведених досліджень було визначено технологічний стек та розроблено архітектуру програмного забезпечення.

Також, було створено ідею стартап-проекту з аналізом потенційних техніко-економічних переваг цієї ідеї та розроблено маркетингову програму стартап проекту.

Матеріали роботи були опубліковані у збірнику тез IV всеукраїнської науково-практичної конференції молодих вчених та студентів «Інформаційні системи та технології управління» (ІСТУ-2020) Худа А. О., Халус О. А. «Метод порівняння аудіофайлів у системі для вивчення іноземних слів» [16].

ПЕРЕЛІК ПОСИЛАНЬ

1) 9 додатків для вивчення правильної вимови англійських слів [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: http://headtechnology.com.ua/120-9-dodatkiv-dlya-vivchennya-pravilnoyi-vimovi-angliyskikh-sliv.html.

2) Müller M. Fundamentals of Music Processing — Audio, Analysis, Algorithms, Applications. / Meinard Müller., 2015. – 480 с.

3) Применение преобразования Фурье в цифровой обработке звука [Електронний ресурс] – Режим доступу до ресурсу: http://shackmaster.narod.ru/fourier.htm.

4) В. П. Денисюк, В. К. Репета. Вища математика – Київ: НАУ, 2013. – 145 с.

5) Schildt H. Java: A Beginner's Guide / Herbert Schildt., 2014. – 720 с. – (6th Edition).

6) Модель-представлення-контролер [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Модель-вид-контролер

7) Walls C. Spring in Action / Craig Walls., 2014. – 624 с.

8) Christian Bauer, Gavin King Hibernate in action – M: Manning Publications, 2005 – 408 p.

9) PostgreSQL [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/PostgreSQL

10) Deep Voice: Real-time Neural Text-to-Speech / Coates Adam, 2017. – 17 с.

11) Kamath U. Deep Learning for NLP and Speech Recognition / U. Kamath, J. Liu, J. Whitaker., 2019. – 640 с. – (1st Edition).

12) Dutta A. Audio Processing and Speech Recognition: Concepts, Techniques and Research Overviews / A. Dutta, S. Sen, N. Dey., 2019. – 107 с. – (1st Edition).

13) Mehta P. Comparative Study of Various Algorithms for Speech Synthesis System / P. Mehta, B. Prajapti., 2016. – 96 с.

14) Dutoit T. An Introduction to Text-to-Speech Synthesis / Thierry Dutoit., 1997. – 330 с.

15) Aggarval C. C. Neural Networks and Deep Learning / Charu C. Aggarval., 2018. – 497 с.

16) Худа А.О., Халус О. А. «Метод порівняння аудіофайлів у системі для вивчення іноземних слів» // Матеріали IV всеукраїнська науково-практична конференція молодих вчених та студентів «Інформаційні системи та технології управління» – м. Київ.: НТУУ «КПІ ім. Ігоря Сікорського»

ДОДАТОК А. Програмний код

Quiz.java

```
package org.kpi.easylearn.model;

import lombok.Data;

import java.util.List;

@Data
public class Quiz {
    List<Word> remainingWords;
    Word currentWord;
    List<Word> options;
    QuizType type;
    List<Word> wrongWords;
    List<Word> correctWords;

    public boolean isFinished() {
        return currentWord == null;
    }
}
```

QuizType.java

```
package org.kpi.easylearn.model;

public enum QuizType {
    BY_WORD,
    BY_TRANSLATION,
    BY_IMAGE
}
```

User.java

```
package org.kpi.easylearn.model;

import lombok.Data;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.NotEmpty;
import javax.validation.constraints.Pattern;
import javax.validation.constraints.Size;

@Entity
@Data
@Table(name = "users")
public class User {
    @Id
    @GeneratedValue
    Long id;
    @Size(min = 4, max = 255)
    String username;
    @Size(max = 255)
    @Pattern(regexp = "[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,4}")
    String email;
    @NotEmpty
    String password;
    boolean isAdmin;
}
```

UserStats.java

```
package org.kpi.easylearn.model;

import lombok.Data;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
import java.sql.Timestamp;
```

```
@Data
@Entity
public class UserStats {
    @Id
    @GeneratedValue
    Long id;
    @ManyToOne(optional = false)
    User user;
    Timestamp time;
    int learningRate;
}
```

VocabularyWord.java

```
package org.kpi.easylearn.model;

import lombok.Data;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
import javax.validation.constraints.Min;
import javax.validation.constraints.Size;

@Entity
@Data
public class VocabularyWord {
    @Id
    @GeneratedValue
    Long id;
    @Size(min = 1, max = 255, message = "Word size is invalid")
    String text;
    @Size(min = 1, max = 255, message = "Translation size is invalid")
    String translation;
}
```

Word.java

```
package org.kpi.easylearn.model;

import lombok.Data;
import lombok.NonNull;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
import javax.validation.constraints.Min;
import javax.validation.constraints.NotEmpty;
import javax.validation.constraints.Size;

@Entity
@Data
public class Word {
    @Id
    @GeneratedValue
    Long id;
    @Size(min = 1, max = 255, message = "Word size is invalid")
    String text;
    @Size(min = 1, max = 255, message = "Translation size is invalid")
    String translation;
    @ManyToOne(optional = false)
    User user;
    @Min(0)
    int correctAnswers;
    @Min(0)
    int wrongAnswers;
}
```

WordImage.java

```
package org.kpi.easylearn.model;

import lombok.Data;
```

```
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.OneToOne;
import javax.validation.constraints.NotEmpty;

@Data
@Entity
public class WordImage {
    @Id
    @GeneratedValue
    Long id;
    @OneToOne(optional = false)
    Word word;
    @NotEmpty
    byte[] imageBytes;
}
```

UserRepository.java

```
package org.kpi.easylearn.repository;

import org.kpi.easylearn.model.User;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.CrudRepository;
import org.springframework.data.repository.query.Param;

public interface UserRepository extends CrudRepository<User, Long> {

    @Query("select u from User u where u.username = :identifier or u.email = :identifier")
    User findByIdentifier(@Param("identifier") String identifier);


}
```

UserStatsRepository.java

```
package org.kpi.easylearn.repository;

import org.kpi.easylearn.model.User;
import org.kpi.easylearn.model.UserStats;
import org.springframework.data.repository.CrudRepository;

import java.util.List;

public interface UserStatsRepository extends CrudRepository<UserStats, Long> {
    List<UserStats> findTop10ByUserOrderByTimeDesc(User user);
}
```

WordImageRepository.java

```
package org.kpi.easylearn.repository;

import org.kpi.easylearn.model.Word;
import org.kpi.easylearn.model.WordImage;
import org.springframework.data.repository.CrudRepository;

public interface WordImageRepository extends CrudRepository<WordImage, Long> {
    WordImage findByWord(Word word);
}
```

WordRepository.java

```
package org.kpi.easylearn.repository;

import org.kpi.easylearn.model.User;
import org.kpi.easylearn.model.Word;
import org.springframework.data.domain.Pageable;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Modifying;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.CrudRepository;
import org.springframework.data.repository.query.Param;
import org.springframework.transaction.annotation.Transactional;

import java.util.List;

public interface WordRepository extends CrudRepository<Word, Long> {
```

```
            List<Word> findByUserOrderByTextAsc(User user, Pageable page);

            int countByUser(User user);

            @Query("select count(w) from WordImage wi join Word w on wi.word = w where w.user = :user")
            int countByUserWithImages(@Param("user") User user);

            @Query("select count(w) from Word w where w.user = :user and w.correctAnswers > w.wrongAnswers
* 2 " +
                    "and w.correctAnswers > 2")
            int countLearnedWords(@Param("user") User user);

            @Modifying
            @Transactional
            @Query("update Word w set w.correctAnswers = w.correctAnswers + 1 where w in (:words)")
            void incrementCorrectAnswers(@Param("words") List<Word> words);

            @Modifying
            @Transactional
            @Query("update Word w set w.wrongAnswers = w.wrongAnswers + 1 where w in (:words)")
            void incrementWrongAnswers(@Param("words") List<Word> words);

            @Query("select w from Word w where w.user = :user order by rand()")
            List<Word> findRandomUserWords(@Param("user") User user, Pageable page);

            @Query("select w from WordImage wi join Word w on wi.word = w where w.user = :user order by
rand()")
            List<Word> findRandomUserWordsWithImages(@Param("user") User user, Pageable page);

            Word findByUserAndText(User user, String text);
        }
```

CurrentUserResolverService.java

```
        package org.kpi.easylearn.service;

        import org.kpi.easylearn.model.User;

        public interface CurrentUserResolverService {
            User getCurrentUser();
        }
```

ImageService.java

```
        package org.kpi.easylearn.service;

        import org.kpi.easylearn.exception.ImageTransformationException;
        import org.springframework.http.MediaType;
        import org.springframework.web.multipart.MultipartFile;

        public interface ImageService {
            byte[] transform(MultipartFile imageFile) throws ImageTransformationException;
            MediaType mediaType();
            byte[] unknownImage();
        }
```

compare.py

```
        from future import print_function
        import numpy as np
        import matplotlib
        import matplotlib.pyplot as plt

        import librosa
        import librosa.display

        D, wp = librosa.core.dtw(X=x_1_chroma, Y=x_2_chroma, metric='cosine')
        wp_s = np.asarray(wp) * hop_size / fs

        def scale_minmax(X, min=0.0, max=1.0):
            X_std = (X - X.min()) / (X.max() - X.min())
            X_scaled = X_std * (max - min) + min
            return X_scaled

        def stft(y, n_fft=2048):
            if win_length is None:
                win_length = n_fft

            if hop_length is None:
```

```
        hop_length = int(win_length // 4)

    fft_window = get_window(window, win_length, fftbins=True)
    fft_window = util.pad_center(fft_window, n_fft)
    fft_window = fft_window.reshape((-1, 1))

    util.valid_audio(y)

    if center:
        y = np.pad(y, int(n_fft // 2), mode=pad_mode)

    y_frames = util.frame(y, frame_length=n_fft, hop_length=hop_length)

    stft_matrix = np.empty((int(1 + n_fft // 2), y_frames.shape[1]),
                           dtype=dtype,
                           order='F')

    n_columns = int(util.MAX_MEM_BLOCK / (stft_matrix.shape[0] *
                                          stft_matrix.itemsize))

    for bl_s in range(0, stft_matrix.shape[1], n_columns):
        bl_t = min(bl_s + n_columns, stft_matrix.shape[1])

        stft_matrix[:, bl_s:bl_t] = fft.fft(fft_window *
                                            y_frames[:, bl_s:bl_t],
                                            axis=0)[:stft_matrix.shape[0]]

x_1, fs = librosa.load(sys.args[1])
x_2, fs = librosa.load(sys.args[2])

n_fft = 4410
hop_size = 2205

x_1_chroma = librosa.feature.chroma_stft(y=x_1, sr=fs, tuning=0, norm=2,
                                         hop_length=hop_size, n_fft=n_fft)
x_2_chroma = librosa.feature.chroma_stft(y=x_2, sr=fs, tuning=0, norm=2,
                                         hop_length=hop_size, n_fft=n_fft)

print(norm(diff(x_1_chroma, x_2_chroma)))
```

QuizService.java

```
package org.kpi.easylearn.service;

import org.kpi.easylearn.model.Quiz;
import org.kpi.easylearn.model.QuizType;
import org.kpi.easylearn.model.User;

public interface QuizService {
    Quiz createQuiz(User user, QuizType quizType);
    void nextStep(User user, Quiz quiz, long selectedWordId);
    boolean isQuizAllowed(User user, QuizType quizType);
}
```

StatsService.java

```
package org.kpi.easylearn.service;

import org.kpi.easylearn.model.User;

import java.util.List;

public interface StatsService {
    List<Integer> getUserLearningRate(User user);

    void recordStats(User user);
}
```

UserService.java

```
package org.kpi.easylearn.service;

import org.kpi.easylearn.exception.RegistrationException;
import org.kpi.easylearn.model.User;

public interface UserService {
    User register(User user) throws RegistrationException;
}
```

WordImageService.java

```
package org.kpi.easylearn.service;

import org.kpi.easylearn.exception.ImageTransformationException;
import org.kpi.easylearn.model.Word;
import org.springframework.http.MediaType;
import org.springframework.web.multipart.MultipartFile;

public interface WordImageService {
    void saveWordImage(Word word, MultipartFile imageFile) throws ImageTransformationException;
    byte[] getWordImageBytes(long wordId);
    MediaType imageMediaType();
    void deleteWordImage(Word word);
}
```

WordService.java

```
package org.kpi.easylearn.service;

import org.kpi.easylearn.exception.ImageTransformationException;
import org.kpi.easylearn.exception.WordException;
import org.kpi.easylearn.model.User;
import org.kpi.easylearn.model.Word;
import org.springframework.web.multipart.MultipartFile;

import java.util.List;

public interface WordService {
    void createWord(Word word, MultipartFile multipartFile) throws WordException,
ImageTransformationException;
    void deleteWord(User user, Long id);
    List<Word> getIndexUserWords(User user, int page);
    List<Word> getRandomUserWords(User user, int count, boolean imageRequired);
    int getPagesCount(User user);
}
```

ImageServiceImpla.java

```
package org.kpi.easylearn.service.impl;

import lombok.AllArgsConstructor;
import lombok.experimental.FieldDefaults;
import org.kpi.easylearn.exception.ImageTransformationException;
import org.kpi.easylearn.service.ImageService;
import org.springframework.core.io.ClassPathResource;
import org.springframework.http.MediaType;
import org.springframework.stereotype.Service;
import org.springframework.web.multipart.MultipartFile;

import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;

import static lombok.AccessLevel.PRIVATE;

@Service
@AllArgsConstructor
@FieldDefaults(level = PRIVATE, makeFinal = true)
public class ImageServiceImpl implements ImageService {

    static final int IMAGE_SIZE = 256;

    static byte[] UNKNOWN_IMAGE_BYTES;
    static {
        byte[] bytes = null;
        try {
            bytes = Files.readAllBytes(Paths.get(new
ClassPathResource("static/img/unknown.png").getURI()));
        } catch (IOException e) {
            throw new RuntimeException("Failed to initialize unknown image bytes", e);
        } finally {
            UNKNOWN_IMAGE_BYTES = bytes;
        }
    }
```

```
            @Override
            public byte[] transform(MultipartFile imageFile) throws ImageTransformationException {
                try {
                    BufferedImage bufferedImage = ImageIO.read(imageFile.getInputStream());
                    if (bufferedImage == null) {
                        throw new ImageTransformationException("Unknown image format");
                    }

                    if (bufferedImage.getHeight() < IMAGE_SIZE || bufferedImage.getWidth() < IMAGE_SIZE)
{
                        throw new ImageTransformationException("Image is too small. Minimum size is " +
IMAGE_SIZE + "x" + IMAGE_SIZE);
                    }

                    if (bufferedImage.getWidth() < bufferedImage.getHeight()) {
                        int cropTop = (bufferedImage.getHeight() - bufferedImage.getWidth()) / 2;
                        bufferedImage = bufferedImage.getSubimage(0, cropTop, bufferedImage.getWidth(),
bufferedImage.getWidth());
                    }

                    if (bufferedImage.getHeight() < bufferedImage.getWidth()) {
                        int cropLeft = (bufferedImage.getWidth() - bufferedImage.getHeight()) / 2;
                        bufferedImage          =          bufferedImage.getSubimage(cropLeft,          0,
bufferedImage.getHeight(), bufferedImage.getHeight());
                    }

                    BufferedImage    resultImage    =    new    BufferedImage(IMAGE_SIZE,    IMAGE_SIZE,
BufferedImage.TYPE_INT_RGB);

                    resultImage.getGraphics().drawImage(bufferedImage,  0,  0,  IMAGE_SIZE,  IMAGE_SIZE,
null);


                    ByteArrayOutputStream out = new ByteArrayOutputStream();
                    ImageIO.write(resultImage, "PNG", out);
                    out.close();
                    return out.toByteArray();
                } catch (IOException e) {
                    throw new RuntimeException(e);
                }
            }

            @Override
            public MediaType mediaType() {
                return MediaType.IMAGE_PNG;
            }

            @Override
            public byte[] unknownImage() {
                return UNKNOWN_IMAGE_BYTES;
            }

        }
```

QuizServiceImpl.java

```
        package org.kpi.easylearn.service.impl;

        import lombok.AllArgsConstructor;
        import lombok.experimental.FieldDefaults;
        import org.kpi.easylearn.common.ListUtil;
        import org.kpi.easylearn.exception.QuizException;
        import org.kpi.easylearn.model.Quiz;
        import org.kpi.easylearn.model.QuizType;
        import org.kpi.easylearn.model.User;
        import org.kpi.easylearn.model.Word;
        import org.kpi.easylearn.repository.WordRepository;
        import org.kpi.easylearn.service.QuizService;
        import org.kpi.easylearn.service.StatsService;
        import org.kpi.easylearn.service.WordService;
        import org.springframework.stereotype.Service;

        import java.util.ArrayList;
        import java.util.List;

        import static lombok.AccessLevel.PRIVATE;

        @Service
```

```
@AllArgsConstructor
@FieldDefaults(level = PRIVATE, makeFinal = true)
public class QuizServiceImpl implements QuizService {

    WordService wordService;
    StatsService statsService;
    WordRepository wordRepository;

    static int QUIZ_WORDS_COUNT = 20;
    static int OPTIONS_COUNT = 4;

    private List<Word> findOptions(User user, Word currentWord, boolean imageRequired) {
        List<Word> options = wordService.getRandomUserWords(user, OPTIONS_COUNT, imageRequired);

        if (options.size() != OPTIONS_COUNT) {
            throw new QuizException("Not enough words");
        }

        options.remove(currentWord);
        options = options.subList(0, Math.min(options.size(), OPTIONS_COUNT - 1));
        ListUtil.randomInsert(options, currentWord);
        return options;
    }

    @Override
    public Quiz createQuiz(User user, QuizType quizType) {

        if (!isQuizAllowed(user, quizType)) {
            throw new QuizException("Quiz is not allowed");
        }

        Quiz quiz = new Quiz();

        quiz.setCorrectWords(new ArrayList<>());
        quiz.setWrongWords(new ArrayList<>());

        boolean imageRequired = quizType == QuizType.BY_IMAGE;

        List<Word> words = wordService.getRandomUserWords(user, QUIZ_WORDS_COUNT, imageRequired);

        Word currentWord = words.get(0);
        words.remove(0);

        quiz.setCurrentWord(currentWord);
        quiz.setRemainingWords(words);
        quiz.setOptions(findOptions(user, currentWord, imageRequired));
        quiz.setType(quizType);

        return quiz;
    }

    @Override
    public void nextStep(User user, Quiz quiz, long selectedWordId) {

        if (quiz.getCurrentWord().getId() == selectedWordId) {
            quiz.getCorrectWords().add(quiz.getCurrentWord());
        } else {
            quiz.getWrongWords().add(quiz.getCurrentWord());
        }

        if (!quiz.getRemainingWords().isEmpty()) {
            Word currentWord = quiz.getRemainingWords().get(0);
            quiz.setCurrentWord(currentWord);
            quiz.getRemainingWords().remove(0);

            quiz.setOptions(findOptions(user,       currentWord,       quiz.getType()       ==
QuizType.BY_IMAGE));
        } else {
            quiz.setCurrentWord(null);
            quiz.setOptions(null);

            if (!quiz.getCorrectWords().isEmpty()) {
                wordRepository.incrementCorrectAnswers(quiz.getCorrectWords());
            }

            if (!quiz.getWrongWords().isEmpty()) {
                wordRepository.incrementWrongAnswers(quiz.getWrongWords());
            }
```

```
                statsService.recordStats(user);
            }
        }

        @Override
        public boolean isQuizAllowed(User user, QuizType quizType) {
            int wordsCount;
            if (quizType == QuizType.BY_TRANSLATION || quizType == QuizType.BY_WORD) {
                wordsCount = wordRepository.countByUser(user);
            } else {
                wordsCount = wordRepository.countByUserWithImages(user);
            }

            return wordsCount >= OPTIONS_COUNT;
        }
    }
```

StatsServiceImpl.java

```
package org.kpi.easylearn.service.impl;

import lombok.AllArgsConstructor;
import lombok.experimental.FieldDefaults;
import org.kpi.easylearn.model.User;
import org.kpi.easylearn.model.UserStats;
import org.kpi.easylearn.repository.UserStatsRepository;
import org.kpi.easylearn.repository.WordRepository;
import org.kpi.easylearn.service.StatsService;
import org.springframework.stereotype.Service;

import java.sql.Timestamp;
import java.util.Collections;
import java.util.List;
import java.util.stream.Collectors;

import static lombok.AccessLevel.PRIVATE;

@Service
@AllArgsConstructor
@FieldDefaults(level = PRIVATE, makeFinal = true)
public class StatsServiceImpl implements StatsService {

    UserStatsRepository userStatsRepository;
    WordRepository wordRepository;

    @Override
    public List<Integer> getUserLearningRate(User user) {
        List<Integer> result = userStatsRepository
            .findTop10ByUserOrderByTimeDesc(user)
            .stream()
            .map(UserStats::getLearningRate)
            .collect(Collectors.toList());
        Collections.reverse(result);
        return result;
    }

    @Override
    public void recordStats(User user) {
        UserStats stats = new UserStats();

        stats.setUser(user);
        stats.setTime(new Timestamp(System.currentTimeMillis()));
        stats.setLearningRate(wordRepository.countLearnedWords(user) * 100 /
wordRepository.countByUser(user));

        userStatsRepository.save(stats);
    }
}
```

UserDetailsServiceImpl.java

```
package org.kpi.easylearn.service.impl;

import lombok.AllArgsConstructor;
import lombok.experimental.FieldDefaults;
import org.kpi.easylearn.exception.NotAuthenticatedException;
import org.kpi.easylearn.model.User;
import org.kpi.easylearn.repository.UserRepository;
import org.kpi.easylearn.service.CurrentUserResolverService;
```

```
import org.springframework.security.authentication.AnonymousAuthenticationToken;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;

import java.util.Collections;

import static lombok.AccessLevel.PRIVATE;

@Service
@AllArgsConstructor
@FieldDefaults(level = PRIVATE, makeFinal = true)
public class UserDetailsServiceImpl implements UserDetailsService, CurrentUserResolverService {

    UserRepository userRepository;

    static String USER_ROLE = "USER";

    private static class UserInfo extends org.springframework.security.core.userdetails.User {

        private User user;

        public UserInfo(User user) {
            super(user.getUsername(),      user.getPassword(),      Collections.singletonList(new
SimpleGrantedAuthority(USER_ROLE)));
            this.user = user;
        }

        public User getUser() {
            return user;
        }
    }

    @Override
    public UserDetails loadUserByUsername(String identifier) throws UsernameNotFoundException {
        User user = userRepository.findByIdentifier(identifier);
        if (user == null) throw new UsernameNotFoundException("User " + identifier + " not found");
        return new UserInfo(user);
    }

    @Override
    public User getCurrentUser() {
        Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
        if (!(authentication instanceof AnonymousAuthenticationToken)) {
            return ((UserDetailsServiceImpl.UserInfo) authentication.getPrincipal()).getUser();
        } else {
            throw new NotAuthenticatedException();
        }
    }
}
```

UserServiceImpl.java

```
package org.kpi.easylearn.service.impl;

import lombok.AllArgsConstructor;
import lombok.experimental.FieldDefaults;
import org.kpi.easylearn.exception.RegistrationException;
import org.kpi.easylearn.model.User;
import org.kpi.easylearn.repository.UserRepository;
import org.kpi.easylearn.service.UserService;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;

import static lombok.AccessLevel.PRIVATE;


@Service
@AllArgsConstructor
@FieldDefaults(level = PRIVATE, makeFinal = true)
public class UserServiceImpl implements UserService {

    UserRepository userRepository;
    PasswordEncoder passwordEncoder;
```

```
    @Override
    public User register(User user) throws RegistrationException {

        if (userRepository.findByIdentifier(user.getUsername()) != null) {
            throw new RegistrationException("User with such username already exists");
        }

        if (userRepository.findByIdentifier(user.getEmail()) != null) {
            throw new RegistrationException("User with such email already exists");
        }

        user.setPassword(passwordEncoder.encode(user.getPassword()));

        return userRepository.save(user);
    }
}
```

WordImageServiceImpl.java

```
package org.kpi.easylearn.service.impl;

import lombok.AllArgsConstructor;
import lombok.experimental.FieldDefaults;
import org.kpi.easylearn.exception.ImageTransformationException;
import org.kpi.easylearn.model.Word;
import org.kpi.easylearn.model.WordImage;
import org.kpi.easylearn.repository.WordImageRepository;
import org.kpi.easylearn.repository.WordRepository;
import org.kpi.easylearn.service.ImageService;
import org.kpi.easylearn.service.WordImageService;
import org.springframework.http.MediaType;
import org.springframework.stereotype.Service;
import org.springframework.web.multipart.MultipartFile;

import java.util.Optional;

import static lombok.AccessLevel.PRIVATE;

@Service
@AllArgsConstructor
@FieldDefaults(level = PRIVATE, makeFinal = true)
public class WordImageServiceImpl implements WordImageService {

    ImageService imageService;
    WordImageRepository wordImageRepository;
    WordRepository wordRepository;

    @Override
    public void saveWordImage(Word word, MultipartFile imageFile) throws
ImageTransformationException {
        if (!imageFile.isEmpty()) {
            WordImage wordImage = new WordImage();
            wordImage.setImageBytes(imageService.transform(imageFile));
            wordImage.setWord(word);

            wordImageRepository.save(wordImage);
        }
    }

    @Override
    public byte[] getWordImageBytes(long wordId) {
        Optional<Word> word = wordRepository.findById(wordId);

        if (!word.isPresent()) {
            return imageService.unknownImage();
        } else {
            WordImage wordImage = wordImageRepository.findByWord(word.get());

            byte[] imageBytes;
            if (wordImage != null) {
                imageBytes = wordImage.getImageBytes();
            } else {
                imageBytes = imageService.unknownImage();
            }

            return imageBytes;
        }
    }
```

```
            @Override
            public MediaType imageMediaType() {
                return imageService.mediaType();
            }

            @Override
            public void deleteWordImage(Word word) {
                WordImage wordImage = wordImageRepository.findByWord(word);
                if (wordImage != null) {
                    wordImageRepository.delete(wordImage);
                }
            }
        }
```

WordServiceImpl.java

```
        package org.kpi.easylearn.service.impl;

        import lombok.AllArgsConstructor;
        import lombok.experimental.FieldDefaults;
        import org.kpi.easylearn.exception.AccessDeniedException;
        import org.kpi.easylearn.exception.ImageTransformationException;
        import org.kpi.easylearn.exception.WordException;
        import org.kpi.easylearn.model.User;
        import org.kpi.easylearn.model.Word;
        import org.kpi.easylearn.repository.WordRepository;
        import org.kpi.easylearn.service.WordImageService;
        import org.kpi.easylearn.service.WordService;
        import org.springframework.data.domain.PageRequest;
        import org.springframework.stereotype.Service;
        import org.springframework.web.multipart.MultipartFile;

        import java.util.List;
        import java.util.Optional;

        import static lombok.AccessLevel.PRIVATE;

        @Service
        @AllArgsConstructor
        @FieldDefaults(level = PRIVATE, makeFinal = true)
        public class WordServiceImpl implements WordService {

            WordRepository wordRepository;
            WordImageService wordImageService;

            static int PAGE_SIZE = 20;

            @Override
            public void createWord(Word word, MultipartFile imageFile) throws WordException,
ImageTransformationException {
                if (wordRepository.findByUserAndText(word.getUser(), word.getText()) != null) {
                    throw new WordException("Such word already exists");
                }

                word = wordRepository.save(word);
                wordImageService.saveWordImage(word, imageFile);
            }

            @Override
            public void deleteWord(User user, Long id) {
                Optional<Word> word = wordRepository.findById(id);
                if (word.isPresent()) {
                    if (!word.get().getUser().getId().equals(user.getId())) {
                        throw new AccessDeniedException("Assess denied! Trying to delete not own word!
User: " + user.getId());
                    }

                    wordImageService.deleteWordImage(word.get());
                    wordRepository.delete(word.get());
                }
            }

            @Override
            public List<Word> getIndexUserWords(User user, int page) {
                return wordRepository.findByUserOrderByTextAsc(user, PageRequest.of(page - 1,
PAGE_SIZE));
            }

            @Override
```

```
            public List<Word> getRandomUserWords(User user, int count, boolean imageRequired) {
                if (imageRequired) {
                    return wordRepository.findRandomUserWordsWithImages(user, PageRequest.of(0, count));
                } else {
                    return wordRepository.findRandomUserWords(user, PageRequest.of(0, count));
                }
            }

            @Override
            public int getPagesCount(User user) {
                return   Math.max(1,   (int)   Math.ceil(wordRepository.countByUser(user)   /   (double)
PAGE_SIZE));
            }

        }
```

ExceptionController.java

```
        package org.kpi.easylearn.controller;

        import lombok.AllArgsConstructor;
        import lombok.experimental.FieldDefaults;
        import org.kpi.easylearn.exception.UserAsseptebleException;
        import org.slf4j.Logger;
        import org.slf4j.LoggerFactory;
        import org.springframework.boot.web.servlet.error.ErrorController;
        import org.springframework.http.HttpStatus;
        import org.springframework.stereotype.Controller;
        import org.springframework.ui.Model;
        import org.springframework.web.bind.annotation.RequestMapping;

        import javax.servlet.http.HttpServletRequest;

        import static lombok.AccessLevel.PRIVATE;

        @Controller
        @AllArgsConstructor
        @FieldDefaults(level = PRIVATE, makeFinal = true)
        public class ExceptionController implements ErrorController {

            Logger log = LoggerFactory.getLogger(this.getClass());

            @RequestMapping("/error")
            public String handleError(HttpServletRequest request, Model model) {
                int errorCode = (Integer) request.getAttribute("javax.servlet.error.status_code");

                if (HttpStatus.valueOf(errorCode).is4xxClientError()) {
                    switch (errorCode) {
                        case 400: {
                            model.addAttribute("error", "400, Bad Request");
                            break;
                        }
                        case 401: {
                            model.addAttribute("error", "401, Unauthorized");
                            break;
                        }
                        case 404: {
                            model.addAttribute("error", "404, Not found");
                            break;
                        }
                    }
                } else {
                    Exception                  exception                  =                  (Exception)
request.getAttribute("javax.servlet.error.exception");

                    if (exception.getCause() instanceof UserAsseptebleException) {
                        model.addAttribute("error", exception.getCause().getMessage());
                        log.warn("User acceptable exception", exception);
                    } else {
                        log.error("Critical exception", exception);
                    }
                }

                return "error";
            }

            @Override
            public String getErrorPath() {
                return "/error";
```

```
    }
}
```

IndexController.java

```
package org.kpi.easylearn.controller;

import lombok.AllArgsConstructor;
import lombok.experimental.FieldDefaults;
import org.kpi.easylearn.common.CurrentUser;
import org.kpi.easylearn.model.QuizType;
import org.kpi.easylearn.model.User;
import org.kpi.easylearn.model.Word;
import org.kpi.easylearn.service.QuizService;
import org.kpi.easylearn.service.StatsService;
import org.kpi.easylearn.service.WordService;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;

import java.util.ArrayList;
import java.util.List;

import static lombok.AccessLevel.PRIVATE;

@Controller
@AllArgsConstructor
@FieldDefaults(level = PRIVATE, makeFinal = true)
public class IndexController {

    WordService wordService;
    StatsService statsService;
    QuizService quizService;

    static int PAGES_TO_SHOW = 5;

    @GetMapping("/")
    String indexPage(@RequestParam(required = false, defaultValue = "1") int page,
                     @CurrentUser User user,
                     Model model) {

        int lastPage = wordService.getPagesCount(user);
        int fromPage = Math.max(1, page - (PAGES_TO_SHOW / 2) + 1);
        int toPage = Math.min(lastPage, fromPage + PAGES_TO_SHOW - 1);
        fromPage = Math.max(1, toPage - PAGES_TO_SHOW + 1);
        List<Integer> pages = new ArrayList<>();
        for (int p = fromPage; p <= toPage; p++) {
            pages.add(p);
        }

        model.addAttribute("lastPage", lastPage);
        model.addAttribute("currentPage", page);
        model.addAttribute("pages", pages);

        model.addAttribute("user", user);
        model.addAttribute("words", wordService.getIndexUserWords(user, page));
        model.addAttribute("word", new Word());
        model.addAttribute("learningRate", statsService.getUserLearningRate(user));

        model.addAttribute("isByWordAllowed",                    quizService.isQuizAllowed(user,
QuizType.BY_WORD));
        model.addAttribute("isByTranslationAllowed",             quizService.isQuizAllowed(user,
QuizType.BY_TRANSLATION));
        model.addAttribute("isByImageAllowed",                   quizService.isQuizAllowed(user,
QuizType.BY_IMAGE));

        return "index";
    }

}
```

QuizController.java

```
package org.kpi.easylearn.controller;
```

```
import lombok.AllArgsConstructor;
import lombok.experimental.FieldDefaults;
import org.kpi.easylearn.common.CurrentUser;
import org.kpi.easylearn.model.Quiz;
import org.kpi.easylearn.model.QuizType;
import org.kpi.easylearn.model.User;
import org.kpi.easylearn.service.QuizService;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.SessionAttribute;

import javax.servlet.http.HttpServletRequest;

import static lombok.AccessLevel.PRIVATE;

@Controller
@AllArgsConstructor
@FieldDefaults(level = PRIVATE, makeFinal = true)
public class QuizController {

    QuizService quizService;

    @GetMapping("/quiz")
    public String quizPage(@SessionAttribute(required = false) Quiz quiz,
                           @CurrentUser User user,
                           Model model) {

        if (quiz == null) {
            return "redirect:/";
        }

        model.addAttribute("user", user);

        int totalWords = quiz.getRemainingWords().size() + quiz.getCorrectWords().size() +
quiz.getWrongWords().size() + 1;
        int wrongPercent = quiz.getWrongWords().size() * 100 / totalWords;
        int correctPercent = quiz.getCorrectWords().size() * 100 / totalWords;

        model.addAttribute("wrong", wrongPercent);
        model.addAttribute("correct", correctPercent);


        if (quiz.getType() == QuizType.BY_TRANSLATION) {
            return "quiz-by-translation";
        } else if (quiz.getType() == QuizType.BY_WORD) {
            return "quiz-by-word";
        } else if (quiz.getType() == QuizType.BY_IMAGE) {
            return "quiz-by-image";
        } else {
            throw new IllegalStateException("Invalid quiz type");
        }
    }

    @GetMapping("/result")
    public String resultPage(@SessionAttribute(required = false) Quiz quiz,
                             @CurrentUser User user,
                             Model model) {

        if (quiz == null) {
            return "redirect:/";
        }

        model.addAttribute("user", user);

        int totalWords = quiz.getCorrectWords().size() + quiz.getWrongWords().size();
        int wrongPercent = quiz.getWrongWords().size() * 100 / totalWords;
        int correctPercent = 100 - wrongPercent;

        model.addAttribute("wrong", wrongPercent);
        model.addAttribute("correct", correctPercent);

        return "result";
    }

    @PostMapping("/quiz/start")
    public String startQuiz(@RequestParam QuizType type,
```

```
                                @CurrentUser User user,
                                HttpServletRequest request) {

            Quiz quiz = quizService.createQuiz(user, type);
            request.getSession().setAttribute("quiz", quiz);

            return "redirect:/quiz";
        }

        @PostMapping("/quiz")
        public String submitAnswer(@RequestParam long answer,
                                   @SessionAttribute Quiz quiz,
                                   @CurrentUser User user,
                                   HttpServletRequest request) {

            quizService.nextStep(user, quiz, answer);
            request.getSession().setAttribute("quiz", quiz);

            if (!quiz.isFinished()) {
                return "redirect:/quiz";
            } else {
                return "redirect:/result";
            }
        }
    }
```

UserController.java

```
    package org.kpi.easylearn.controller;

    import lombok.AllArgsConstructor;
    import lombok.experimental.FieldDefaults;
    import org.kpi.easylearn.exception.RegistrationException;
    import org.kpi.easylearn.model.User;
    import org.kpi.easylearn.model.dto.RegistrationUser;
    import org.kpi.easylearn.service.UserService;
    import org.springframework.stereotype.Controller;
    import org.springframework.ui.Model;
    import org.springframework.validation.BindingResult;
    import org.springframework.web.bind.annotation.GetMapping;
    import org.springframework.web.bind.annotation.ModelAttribute;
    import org.springframework.web.bind.annotation.PostMapping;

    import javax.validation.Valid;

    import static lombok.AccessLevel.PRIVATE;

    @Controller
    @AllArgsConstructor
    @FieldDefaults(level = PRIVATE, makeFinal = true)
    public class UserController {

        UserService userService;

        @GetMapping("/register")
        public String registerPage(Model model) {
            model.addAttribute("user", new RegistrationUser());
            return "register";
        }

        @PostMapping("/register")
        public String register(@ModelAttribute @Valid RegistrationUser user, BindingResult
bindingResult, Model model) {
            if (bindingResult.hasErrors()) {
                model.addAttribute("error",
bindingResult.getAllErrors().get(0).getDefaultMessage());
                model.addAttribute("user", user);
                return "register";
            }

            User u = new User();
            u.setUsername(user.getUsername());
            u.setEmail(user.getEmail());
            u.setPassword(user.getPassword());

            try {
                userService.register(u);
            } catch (RegistrationException e) {
                model.addAttribute("error", e.getMessage());
```

```
                model.addAttribute("user", user);
                return "register";
            }

            return "redirect:login";
        }

        @GetMapping("/login")
        public String loginPage() {
            return "login";
        }

    }
```

WordController.java

```
    package org.kpi.easylearn.controller;


    import lombok.AllArgsConstructor;
    import lombok.experimental.FieldDefaults;
    import org.kpi.easylearn.common.CurrentUser;
    import org.kpi.easylearn.exception.ImageTransformationException;
    import org.kpi.easylearn.exception.WordException;
    import org.kpi.easylearn.model.User;
    import org.kpi.easylearn.model.Word;
    import org.kpi.easylearn.service.WordImageService;
    import org.kpi.easylearn.service.WordService;
    import org.springframework.http.HttpEntity;
    import org.springframework.http.HttpHeaders;
    import org.springframework.stereotype.Controller;
    import org.springframework.ui.Model;
    import org.springframework.validation.BindingResult;
    import org.springframework.web.bind.annotation.*;
    import org.springframework.web.multipart.MultipartFile;

    import javax.validation.Valid;

    import static lombok.AccessLevel.PRIVATE;

    @Controller
    @RequestMapping("/word")
    @AllArgsConstructor
    @FieldDefaults(level = PRIVATE, makeFinal = true)
    public class WordController {

        WordService wordService;
        WordImageService wordImageService;

        @PostMapping("/create")
        public String createWord(@ModelAttribute @Valid Word word,
                                 BindingResult bindingResult,
                                 @RequestParam MultipartFile imageFile,
                                 @CurrentUser User user) {

            if (bindingResult.hasErrors()) {
                return "redirect:/?error=" + bindingResult.getAllErrors().get(0).getDefaultMessage()
+ "#dictionary";
            }

            try {
                word.setUser(user);
                wordService.createWord(word, imageFile);
                return "redirect:/#dictionary";
            } catch (WordException | ImageTransformationException e) {
                return "redirect:/?error=" + e.getMessage() + "#dictionary";
            }
        }

        @PostMapping("/delete")
        public String deleteWord(@RequestParam long id,
                                 @CurrentUser User user) {
            wordService.deleteWord(user, id);
            return "redirect:/#dictionary";
        }
```

```
    @GetMapping("/{id}/img")
    @ResponseBody
    public HttpEntity<byte[]> image(@PathVariable Long id) {
        HttpHeaders headers = new HttpHeaders();
        headers.setContentType(wordImageService.imageMediaType());
        byte[] imageBytes = wordImageService.getWordImageBytes(id);
        headers.setContentLength(imageBytes.length);
        return new HttpEntity<>(imageBytes, headers);
    }

}
```

ДОДАТОК Б. Графічний матеріал

Схема структурна бізнес-процесу визначення правильності вимови.

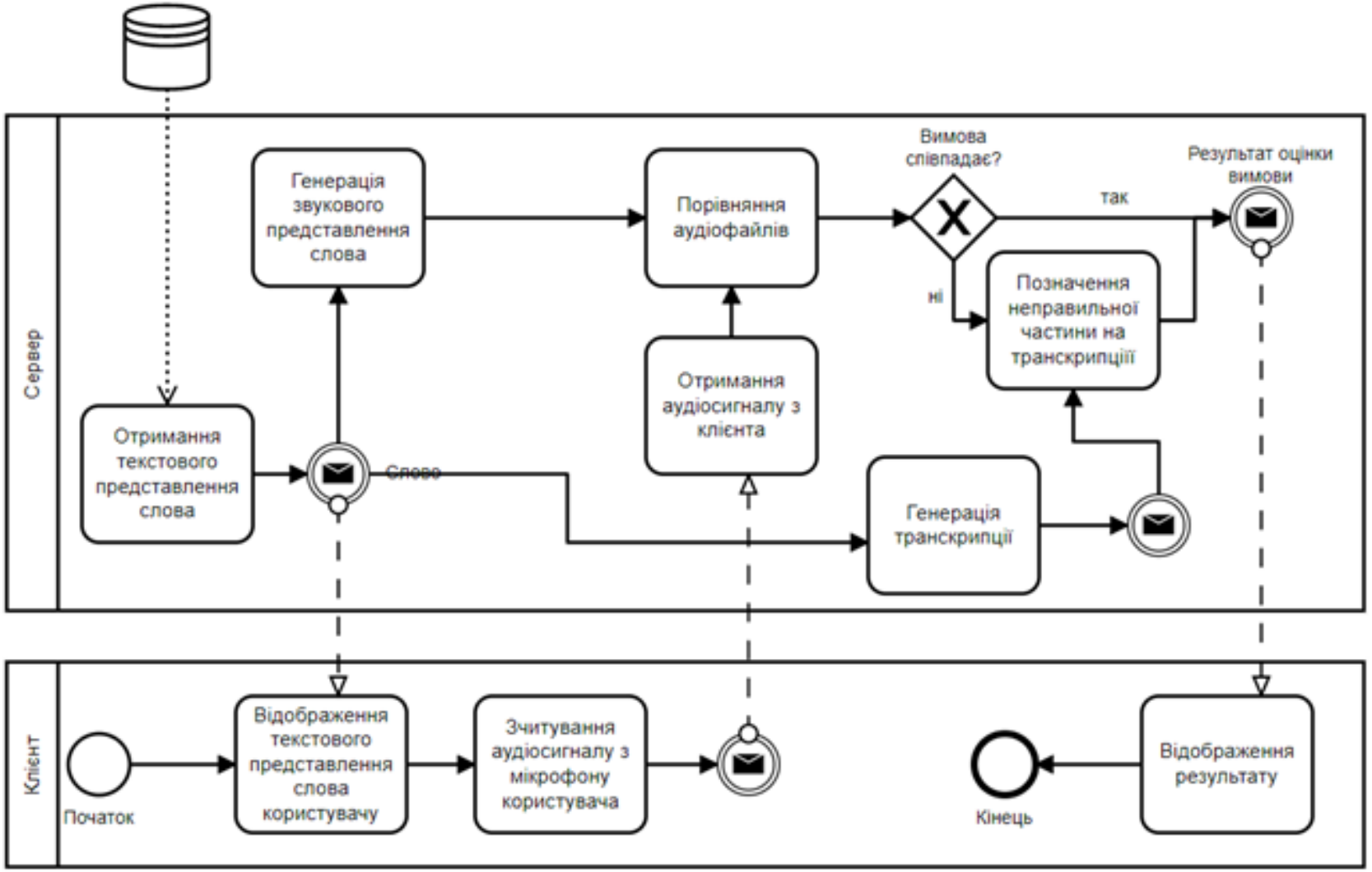
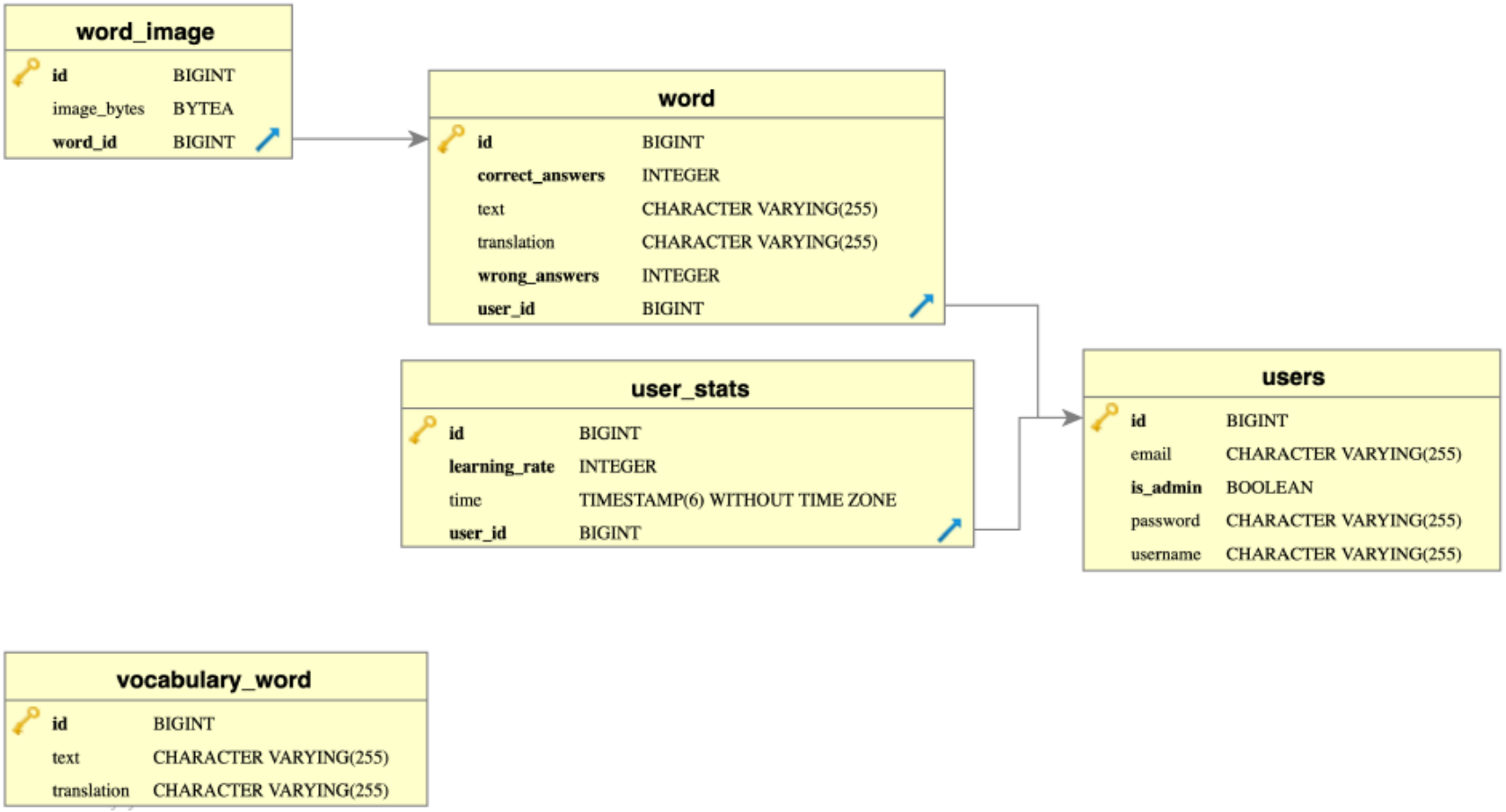


Схема структурна бази даних



UNICHECK

Ім'я користувача:
Попенко Володимир Дмитрович

ID перевірки:
1005447532

Дата перевірки:
14.12.2020 00:12:39 EET

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
14.12.2020 00:39:15 EET

ID користувача:
77149

Назва документа: **Huda_magistr_ip92mp_2**

Кількість сторінок: **54** Кількість слів: **9478** Кількість символів: **74645** Розмір файлу: **1.47 MB** ID файлу: **1005737886**

10.6%
Схожість

Найбільша схожість: **1.87%** з джерелом з Бібліотеки (ID файлу: **1000037865**)

5.47% Джерела з Інтернету	251	Сторінка 56
9.33% Джерела з Бібліотеки	470	Сторінка 57

0.09% Цитат

Цитати	1	Сторінка 58

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	13